IBM WebSphere MQ

Installing IBM WebSphere MQ

Version 7 Release 5

Note

Before using this information and the product it supports, read the information in "Notices" on page 1193.

This edition applies to version 7 release 5 of WebSphere MQ and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Configuring 737

Figures

Tables

Planning

When planning your IBM® WebSphere® MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Before you plan your IBM WebSphere MQ environment, familiarize yourself with the basic IBM WebSphere MQ concepts, see the topics in Technical overview.

Related information:

Migrating

Installing
Before you start installing IBM WebSphere MQ, consider how you want to use it. Use these topics to help you to prepare for installation, install the product, and verify the installation. There is also information to help you to uninstall the product.

Configuring
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

Administering WebSphere MQ

Making sure that messages are not lost (logging)
WebSphere MQ logs all the information you need to recover from a queue manager failure.

Availability, recovery and restart
Make your applications highly available by maintaining queue availability if a queue manager fails, and recover messages after server or storage failure.

Designing a IBM WebSphere MQ architecture

Find out about the different architectures that IBM WebSphere MQ supports for point-to-point and publish/subscribe messaging styles.

Before you plan your IBM WebSphere MQ architecture, familiarize yourself with the basic IBM WebSphere MQ concepts, see the topics in IBM WebSphere MQ Technical overview.

IBM WebSphere MQ architectures range from simple architectures using a single queue manager, to more complex networks of interconnected queue managers. Multiple queue managers are connected together using distributed queueing techniques. For more information about planning single queue manager and multiple queue manager architectures, see the following topics:

- "Architectures based on a single queue manager" on page 2
- "Architectures based on multiple queue managers" on page 3
- "Networks and Network Planning" on page 3
- WebSphere MQ distributed-messaging techniques

If you need multiple queue managers that are logically related and need to share data and applications they can be grouped together in a cluster. Using clusters can enable queue managers to communicate with each other without the need to set up extra channel definitions or remote queue definitions, simplifying their configuration and management. For more information about using clusters, see How clusters work.

Related concepts:

"Planning" on page 1
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Related information:

Configuring
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

Architectures based on a single queue manager

The simplest IBM WebSphere MQ architectures involve the configuration and use of a single queue manager.

Before you plan your IBM WebSphere MQ architecture, familiarize yourself with the basic IBM WebSphere MQ concepts, see Introduction to IBM WebSphere MQ.

A number of possible architectures using a single queue manager are described in the following sections:

- "Single queue manager with local applications accessing a service"
- "Single queue manager with remote applications accessing a service as clients"
- "Single queue manager with a publish/subscribe configuration"

Single queue manager with local applications accessing a service

The first architecture based on a single queue manager is where the applications accessing a service are running on the same system as the applications providing the service. A IBM WebSphere MQ queue manager provides asynchronous intercommunication between the applications requesting the service and the applications providing the service. This means that communication between the applications can continue even if one of the applications is offline for an extended period of time.

Single queue manager with remote applications accessing a service as clients

The second architecture based on a single queue manager has the applications running remotely from the applications providing the service. The remote applications are running on different systems to the services. The applications connect as clients to the single queue manager. This means that access to a service can be provided to multiple systems through a single queue manager.

A limitation of this architecture is that a network connection must be available for an application to operate. The interaction between the application and the queue manager over the network connection is synchronous.

Single queue manager with a publish/subscribe configuration

An alternative architecture using a single queue manager is to use a publish/subscribe configuration. In publish/subscribe messaging, you can decouple the provider of information from the consumers of that information. This differs from the point to point messaging styles in the previously described architectures, where the applications must know information about the target application, for example the queue name to put messages on. Using IBM WebSphere MQ publish/subscribe the sending application publishes a message with a specified topic based on the subject of the information. IBM WebSphere MQ handles the distribution of the message to applications that have registered an interest in that subject through a subscription. The receiving applications also do not need to know anything about the source of the messages to receive them. For more information about publish/subscribe messaging, see Introduction to WebSphere MQ publish/subscribe messaging. For an example of publish/subscribe messaging using a single queue manager, see Example of a single queue manager publish/subscribe configuration.

Related concepts:

"Designing a IBM WebSphere MQ architecture" on page 1
Find out about the different architectures that IBM WebSphere MQ supports for point-to-point and publish/subscribe messaging styles.

Related information:

Introduction to WebSphere MQ

Creating and managing queue managers
Before you can use messages and queues, you must create and start at least one queue manager and its associated objects.

Architectures based on multiple queue managers

You can use distributed message queuing techniques to create a IBM WebSphere MQ architecture involving the configuration and use of multiple queue managers.

Before you plan your IBM WebSphere MQ architecture, familiarize yourself with the basic IBM WebSphere MQ concepts, see Introduction to IBM WebSphere MQ.

A IBM WebSphere MQ architecture can be changed, without alteration to applications that provide services, by adding additional queue managers.

Applications can be hosted on the same machine as a queue manager, and then gain asynchronous communication with a service hosted on another queue manager on another system. Alternatively, applications accessing a service can connect as clients to a queue manager that then provides asynchronous access to the service on another queue manager.

Routes that connect different queue managers and their queues are defined using distributed queuing techniques. The queue managers within the architecture are connected using channels. Channels are used to move messages automatically from one queue manager to another in one direction depending on the configuration of the queue managers.

For a high level overview of planning a IBM WebSphere MQ network, see "Networks and Network Planning."

For information about how to plan channels for your IBM WebSphere MQ architecture, see WebSphere MQ distributed-messaging techniques.

Distributed queue management enables you to create and monitor the communication between queue managers. For more information about distributed queue management, see Introduction to distributed queue management.

Related concepts:

"Designing a IBM WebSphere MQ architecture" on page 1
Find out about the different architectures that IBM WebSphere MQ supports for point-to-point and publish/subscribe messaging styles.

Related information:

Introduction to WebSphere MQ

Creating and managing queue managers
Before you can use messages and queues, you must create and start at least one queue manager and its associated objects.

Networks and Network Planning

WebSphere MQ sends and receives data between applications, and over networks using Queue Managers and Channels. Network planning involves defining requirements to create a framework for connecting these systems over a network.

Channels can be created between your system and any other system with which you need to have communications. Multi-hop channels can be created to connect to systems where you have no direct connections. The message channel connections described in the scenarios are shown as a network diagram in Figure 1.

Channel and transmission queue names

Transmission queues can be given any name. But to avoid confusion, you can give them the same names as the destination queue manager names, or queue manager alias names, as appropriate. This associates the transmission queue with the route they use, giving a clear overview of parallel routes created through intermediate (multi-hopped) queue managers.

It is not so clear-cut for channel names. The channel names in Figure 1 for QM2, for example, must be different for incoming and outgoing channels. All channel names can still contain their transmission queue names, but they must be qualified to make them unique.

For example, at QM2, there is a QM3 channel coming from QM1, and a QM3 channel going to QM3. To make the names unique, the first one might be named 'QM3_from_QM1', and the second might be named 'QM3_from_QM2'. In this way, the channel names show the transmission queue name in the first part of the name. The direction and adjacent queue manager name are shown in the second part of the name.

A table of suggested channel names for Figure 1 is given in Table 1.

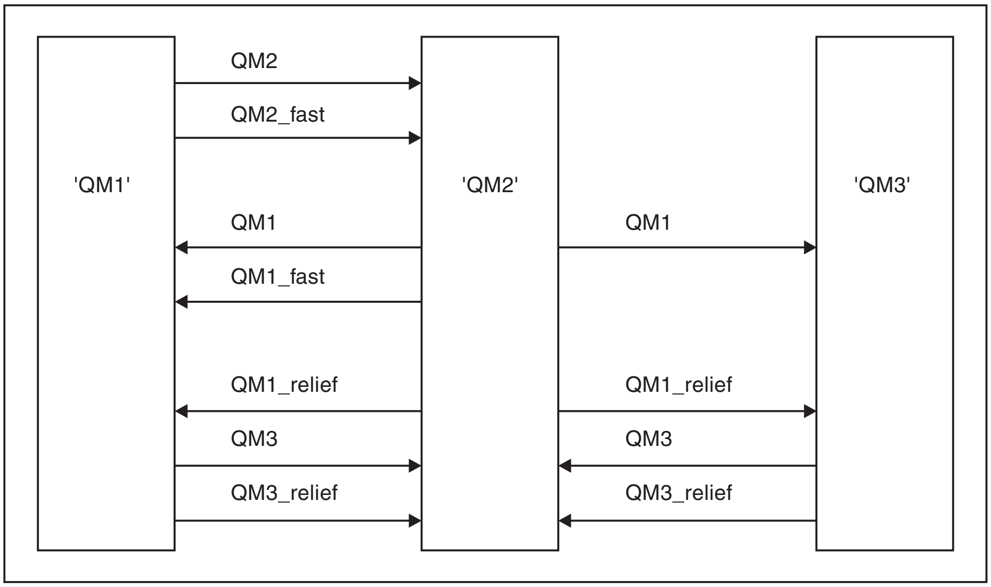


Figure 1. Network diagram showing all channels

Table 1. Example of channel names

Route name	Queue managers hosting channel	Transmission queue name	Suggested channel name
QM1	QM1 & QM2	QM1 (at QM2)	QM1.from.QM2
QM1	QM2 & QM3	QM1 (at QM3)	QM1.from.QM3
QM1_fast	QM1 & QM2	QM1_fast (at QM2)	QM1_fast.from.QM2
QM1_relief	QM1 & QM2	QM1_relief (at QM2)	QM1_relief.from.QM2
QM1_relief	QM2 & QM3	QM1_relief (at QM3)	QM1_relief.from.QM3

Table 1. Example of channel names (continued)

Route name	Queue managers hosting channel	Transmission queue name	Suggested channel name
QM2	QM1 & QM2	QM2 (at QM1)	QM2.from.QM1
QM2_fast	QM1 & QM2	QM2_fast (at QM1)	QM2_fast.from.QM1
QM3	QM1 & QM2	QM3 (at QM1)	QM3.from.QM1
QM3	QM2 & QM3	QM3 (at QM2)	QM3.from.QM2
QM3_relief	QM1 & QM2	QM3_relief (at QM1)	QM3_relief.from.QM1
QM3_relief	QM2 & QM3	QM3_relief (at QM2)	QM3_relief.from.QM2

Note:

1. On WebSphere MQ for z/OS®, queue manager names are limited to four characters.
2. Name all the channels in your network uniquely. As shown in Table 1 on page 4, including the source and target queue manager names in the channel name is a good way to do so.

Network planner

Creating a network assumes that there is another, higher level function of *network planner* whose plans are implemented by the other members of the team.

For widely used applications, it is more economical to think in terms of local access sites for the concentration of message traffic, using wide-band links between the local access sites, as shown in Figure 2 on page 6.

In this example there are two main systems and a number of satellite systems. Tthe actual configuration would depend on business considerations. There are two concentrator queue managers located at convenient centers. Each QM-concentrator has message channels to the local queue managers:

- QM-concentrator 1 has message channels to each of the three local queue managers, QM1, QM2, and QM3. The applications using these queue managers can communicate with each other through the QM-concentrators.
- QM-concentrator 2 has message channels to each of the three local queue managers, QM4, QM5, and QM6. The applications using these queue managers can communicate with each other through the QM-concentrators.
- The QM-concentrators have message channels between themselves thus allowing any application at a queue manager to exchange messages with any other application at another queue manager.

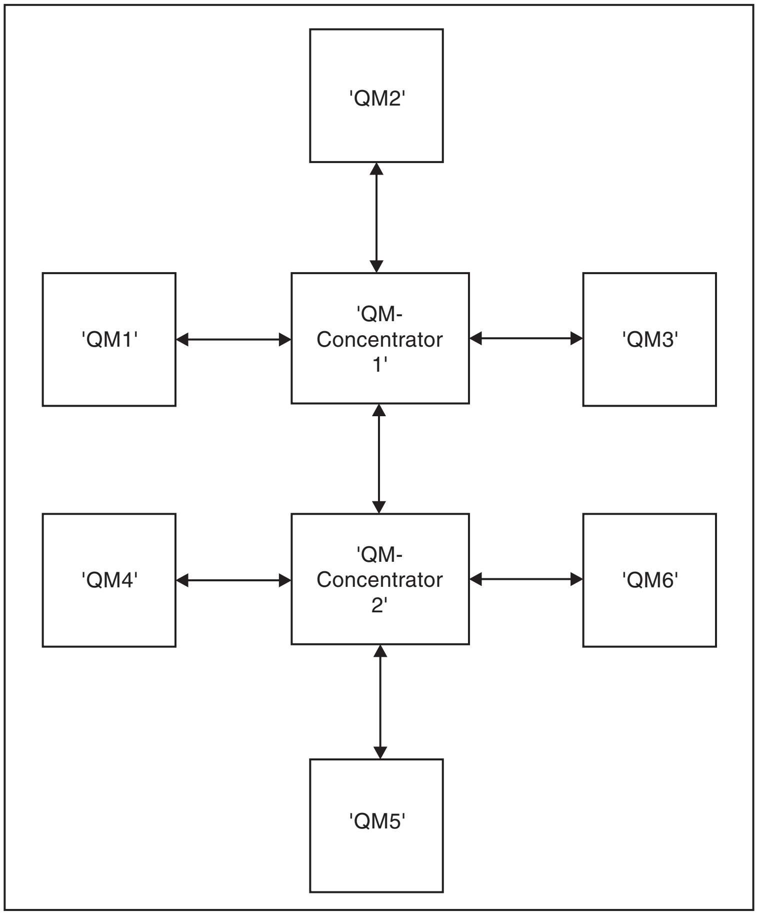


Figure 2. Network diagram showing QM-concentrators

Clustering

This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

The following information assumes that the user has a basic understanding of IBM WebSphere MQ clusters. This information is not intended as a “one size fits all” solution, but is instead trying to share common approaches to common problems.

Clusters provide a mechanism for interconnecting queue managers in a way that simplifies the initial configuration required to set up the system and the ongoing management required. The larger the configuration, the greater the benefit.

Care is required in planning clustering systems to ensure that they function correctly and to ensure the levels of availability and responsiveness required by the system, especially for larger or more complex clustered systems.

A successful cluster setup is dependent on good planning and a thorough understanding of IBM WebSphere MQ fundamentals, such as good application management and network design. Ensure that you are familiar with the information in Concepts of intercommunication and How clusters work .

What are clusters and why are they used?

Clustering provides two key benefits:

- Clusters simplify the administration of IBM WebSphere MQ networks which usually require many object definitions for channels, transmit queues, and remote queues to be configured. This situation is especially true in large, potentially changing, networks where many queue managers need to be interconnected. This architecture is particularly hard to configure and actively maintain.
- Clusters can be used to distribute the workload of message traffic across queues and queue managers in the cluster. Such distribution allows the message workload of a single queue to be distributed across equivalent instances of that queue located on multiple queue managers. This distribution of the workload can be used to achieve greater resilience to system failures, and to improve the scaling performance of particularly active message flows in a system. In such an environment, each of the instances of the distributed queues have consuming applications processing the messages.

Related information:

Clustering: Best practices
Clusters provide a mechanism for interconnecting queue managers. The best practices described in this section are based on testing and feedback from customers.

Point-to-point messaging

The simplest form of messaging in IBM WebSphere MQ is point-to-point messaging.

In point-to-point messaging, a sending application must know information about the receiving application before it can send a message to that application. For example, the sending application might need to know the name of the queue to which to send the information, and might also specify a queue manager name.

An alternative messaging style that you can use with IBM WebSphere MQ is publish/subscribe messaging. Publish/subscribe messaging allows you to decouple the provider of information, from the consumers of that information. The sending application and receiving application do not need to know anything about each other for the information to be sent and received. For more information about publish/subscribe messaging, see Introduction to WebSphere MQ publish/subscribe messaging.

Related information:

Developing applications

WebSphere MQ messages

Introduction to IBM WebSphere MQ publish/subscribe messaging

Publish/subscribe messaging allows you to decouple the provider of information, from the consumers of that information. The sending application and receiving application do not need to know anything about each other for the information to be sent and received.

Before a point-to-point IBM WebSphere MQ application can send a message to another application, it needs to know something about that application. For example, it needs to know the name of the queue to which to send the information, and might also specify a queue manager name.

IBM WebSphere MQ publish/subscribe removes the need for your application to know anything about the target application. All the sending application has to do, is put a IBM WebSphere MQ message, containing the information that it wants, and assign it a topic, that denotes the subject of the information, and let IBM WebSphere MQ handle the distribution of that information. Similarly, the target application does not have to know anything about the source of the information it receives.

Figure 3 on page 8 shows the simplest publish/subscribe system. There is one publisher, one queue manager, and one subscriber. A subscription is sent from the subscriber to the queue manager, a publication is sent from the publisher to the queue manager, and the publication is then forwarded by the queue manager to the subscriber.

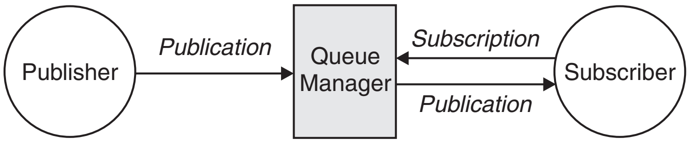


Figure 3. Simple publish/subscribe configuration

A typical publish/subscribe system has more than one publisher and more than one subscriber, and often, more than one queue manager. An application can be both a publisher and a subscriber.

Overview of publish/subscribe components

Publish/subscribe is the mechanism by which subscribers can receive information, in the form of messages, from publishers. The interactions between publishers and subscribers are controlled by queue managers, using standard WebSphere MQ facilities.

A typical publish/subscribe system has more than one publisher and more than one subscriber, and often, more than one queue manager. An application can be both a publisher and a subscriber.

The provider of information is called a *publisher*. Publishers supply information about a subject, without needing to know anything about the applications that are interested in that information. Publishers generate this information in the form of messages, called *publications* that they want to publish and define the topic of these messages.

The consumer of the information is called a *subscriber*. Subscribers create *subscriptions* that describe the topic that the subscriber is interested in. Thus, the subscription determines which publications are forwarded to the subscriber. Subscribers can make multiple subscriptions and can receive information from many different publishers.

Published information is sent in a WebSphere MQ message, and the subject of the information is identified by its *topic*. The publisher specifies the topic when it publishes the information, and the subscriber specifies the topics about which it wants to receive publications. The subscriber is sent information about only those topics it subscribes to.

It is the existence of topics that allows the providers and consumers of information to be decoupled in publish/subscribe messaging by removing the need to include a specific destination in each message as is required in point-to-point messaging.

Interactions between publishers and subscribers are all controlled by a queue manager. The queue manager receives messages from publishers, and subscriptions from subscribers (to a range of topics). The queue manager's job is to route the published messages to the subscribers that have registered an interest in the topic of the messages.

Standard WebSphere MQ facilities are used to distribute messages, so your applications can use all the features that are available to existing WebSphere MQ applications. This means that you can use persistent messages to get once-only assured delivery, and that your messages can be part of a transactional unit-of-work to ensure that messages are delivered to the subscriber only if they are committed by the publisher.

Example of a single queue manager publish/subscribe configuration

Figure 4 illustrates a basic single queue manager publish/subscribe configuration. The example shows the configuration for a news service, where information is available from publishers about several topics:

- Publisher 1 is publishing information about sports results using a topic of Sport
- Publisher 2 is publishing information about stock prices using a topic of Stock
- Publisher 3 is publishing information about film reviews using a topic of Films, and about television listings using a topic of TV

Three subscribers have registered an interest in different topics, so the queue manager sends them the information that they are interested in:

- Subscriber 1 receives the sports results and stock prices
- Subscriber 2 receives the film reviews
- Subscriber 3 receives the sports results

None of the subscribers have registered an interest in the television listings, so these are not distributed.

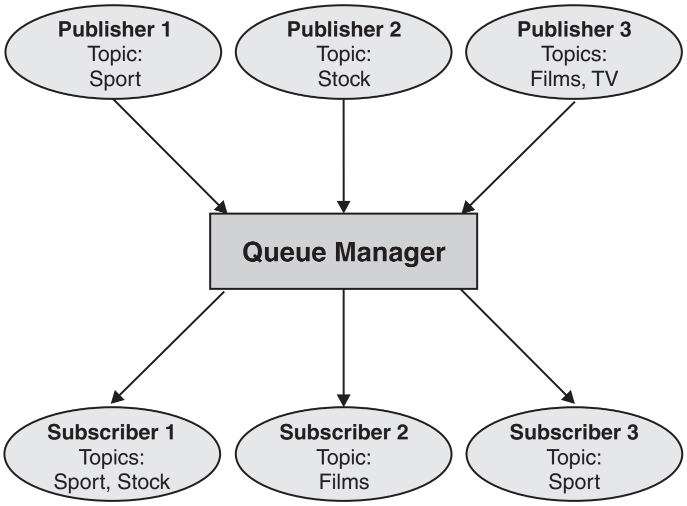


Figure 4. Single queue manager publish/subscribe example. This shows the relationship between publishers, subscribers, and queue managers.

Publishers and publications

In WebSphere MQ publish/subscribe a publisher is an application that makes information about a specified topic available to a queue manager in the form of a standard WebSphere MQ message called a publication. A publisher can publish information about more than one topic.

Publishers use the MQPUT verb to put a message to a previously opened topic, this message is a publication. The local queue manager then routes the publication to any subscribers who have subscriptions to the topic of the publication. A published message can be consumed by more than one subscriber.

In addition to distributing publications to all local subscribers that have appropriate subscriptions, a queue manager can also distribute the publication to any other queue managers connected to it, either directly or through a network of queue managers that have subscribers to the topic.

In a WebSphere MQ publish/subscribe network, a publishing application can also be a subscriber.

Publications under syncpoint

Publishers can issue MQPUT or MQPUT1 calls in syncpoint to include all messages delivered to subscribers in a unit of work. If the MQPMO_RETAIN option, or topic delivery options NPMSGDLV or PMSGDLV with values ALL or ALLDUR are specified, the queue manager uses internal MQPUT or MQPUT1 calls in syncpoint, within the scope of the publisher MQPUT or MQPUT1 call.

State and event information:

Publications can be categorized as either state publications, such as the current price of a stock, or event publications, such as a trade in that stock.

State publications

State publications contain information about the current state of something, such as the price of stock or the current score in a soccer match. When something happens (for example, the stock price changes or the soccer score changes), the previous state information is no longer required because it is superseded by the new information.

A subscriber will want to receive the current version of the state information when it starts, and be sent new information whenever the state changes.

If a publication contains state information, it is often published as a retained publication. A new subscriber typically wants the current state information immediately; the subscriber does not want to wait for an event that causes the information to be republished. Subscribers will automatically receive a topic's retained publication when it subscribes unless the subscriber uses the MQSO_PUBLICATIONS_ON_REQUEST or MQSO_NEW_PUBLICATIONS_ONLY options.

Event publications

Event publications contain information about individual events that occur, such as a trade in some stock or the scoring of a particular goal. Each event is independent of other events.

A subscriber will want to receive information about events as they happen.

Retained publications:

By default, after a publication is sent to all interested subscribers it is discarded. However, a publisher can specify that a copy of a publication is retained so that it can be sent to future subscribers who register an interest in the topic.

Deleting publications after they have been sent to all interested subscribers is suitable for event information, but is not always suitable for state information. By retaining a message, new subscribers do not have to wait for information to be published again before they receive initial state information. For example, a subscriber with a subscription to a stock price would receive the current price straight away, without waiting for the stock price to change (and hence be republished).

The queue manager can retain only one publication for each topic, so the existing retained publication of a topic is deleted when a new retained publication arrives at the queue manager. However, the deletion of the existing publication might not occur synchronously with the arrival of the new retained publication. Therefore, wherever possible, have no more than one publisher sending retained publications on any topic.

Subscribers can specify that they do not want to receive retained publications by using the MQSO_NEW_PUBLICATIONS_ONLY subscription option. Existing subscribers can ask for duplicate copies of retained publications to be sent to them.

There are times when you might not want to retain publications, even for state information:

- If all subscriptions to a topic are made before any publications are made on that topic, and you do not expect, or do not allow, new subscriptions, there is no need to retain publications because they are delivered to the complete set of subscribers the first time they are published.
- If publications occur frequently, such as every second, a new subscriber (or a subscriber recovering from a failure) receives the current state almost immediately after their initial subscription, so there is no need to retain these publications.
- If the publications are large, you might end up needing a considerable amount of storage space to store the retained publication for each topic. In a multiple queue manager environment, retained publications are stored by all queue managers in the network that have a matching subscription.

When deciding whether to use retained publications, consider how subscribing applications recover from a failure. If the publisher does not use retained publications, the subscriber application might need to store its current state locally.

To ensure that a publication is retained, use the MQPMO_RETAIN put-message option. If this option is used and the publication cannot be retained, the message is not published and the call fails with MQRC_PUT_NOT_RETAINED.

If a message is a retained publication, this is indicated by the MQIsRetained message property. The persistence of a message is as it was when it was originally published.

Publications under syncpoint:

In IBM WebSphere MQ publish/subscribe, syncpoint can be used by publishers or internally by the queue manager.

Publishers use syncpoint when they issue MQPUT/MQPUT1 calls with the MQPMO_SYNCPOINT option. All messages delivered to subscribers count towards the maximum number of uncommitted messages in a unit of work.The MAXUMSGS queue manager attribute specifies this limit. If the limit is reached then the publisher receives the 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_REACHED reason code.

When a publisher issues MQPUT/MQPUT1 calls using MQPMO_NO_SYNCPOINT with the MQPMO_RETAIN option, or topic delivery options NPMSGDLV/PMSGDLV with values ALL or ALLDUR, the queue manager uses internal syncpoints to guarantee that messages are delivered as requested. The publisher can receive the 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_REACHED reason code if the limit is reached within the scope of the publisher MQPUT/MQPUT1 call.

Subscribers and subscriptions

In WebSphere MQ publish/subscribe, a subscriber is an application that requests information about a specific topic from a queue manager in a publish/subscribe network. A subscriber can receive messages, about the same or different topics, from more than one publisher.

Subscriptions can be created manually using an MQSC command or by applications. These subscriptions are issued to the local queue manager and contain information about the publications the subscriber wants to receive:

- The topic the subscriber is interested in; this can resolve to multiple topics if wildcards are used.
- An optional selection string to be applied to published messages.
- A handle to a queue (known as the *subscriber queue*), on which selected publications should be placed, and the optional CorrelId.

The local queue manager stores subscription information and when it receives a publication, scans the information to determine whether there is a subscription that matches the publication's topic and selection string. For each matching subscription, the queue manager directs the publication to the subscriber's subscriber queue. The information that a queue manager stores about subscriptions can be viewed by using the DIS SUB and DIS SBSTATUS commands.

A subscription is deleted only when one of the following events occurs:

- The subscriber unsubscribes using the MQCLOSE call (if the subscription was made non-durably).
- The subscription expires.
- The subscription is deleted by the system administrator using the DELETE SUB command.
- The subscriber application ends (if the subscription was made non-durably).
- The queue manager is stopped or restarted (if the subscription was made non-durably).

Managed queues and publish/subscribe:

When you create a subscription you can choose to use managed queuing. If you use managed queueing a subscription queue is automatically created when you create a subscription. Managed queues are tidied up automatically in accordance with the durability of the subscription. Using managed queues means that you do not have to worry about creating queues to receive publications and any unconsumed publications are removed from subscriber queues automatically if a non-durable subscription connection is closed.

If an application has no need to use a particular queue as its subscriber queue, the destination for the publications it receives, it can make use of the *managed subscriptions* using the MQSO_MANAGED subscription option. If you create a managed subscription, the queue manager returns an object handle to the subscriber for a subscriber queue that the queue manager creates where publications will be received. The queue's object handle will be returned allowing you to browse, get or inquire on the queue (it is not possible to put to or set attributes of a managed queue unless you have been explicitly given access to temporary dynamic queues).

The durability of the subscription determines whether the managed queue remains when the subscribing application's connection to the queue manager is broken.

Managed subscriptions are particularly useful when used with non-durable subscriptions because when the application's connection is ended, unconsumed messages would otherwise remain on the subscriber queue taking up space in your queue manager indefinitely. If you are using a managed subscription, the managed queue will be a temporary dynamic queue and as such will be deleted along with any unconsumed messages when the connection is broken for any of the following reasons:

- MQCLOSE with MQCO_REMOVE_SUB is used and the managed Hobj is closed.
- a connection is lost to an application using a non-durable subscription (MQSO_NON_DURABLE).
- a subscription is removed because it has expired and the managed Hobj is closed.

Managed subscriptions can also be used with durable subscriptions but it is possible that you would want to leave unconsumed messages on the subscriber queue so that they can be retrieved when the connection is reopened. For this reason, managed queues for durable subscriptions take the form of a permanent dynamic queue and will remain when the subscribing application's connection to the queue manager is broken.

You can set an expiry on your subscription if you want to use permanent dynamic managed queue so that although the queue will still exist after the connection is broken, it will not continue to exist indefinitely.

If you delete the managed queue you will receive an error message.

The managed queues that are created are named with numbers at the end (timestamps) so that each is unique.

Subscription durability:

Subscriptions can be configured to be durable or non-durable. Subscription durability determines what happens to subscriptions when subscribing applications disconnect from a queue manager.

Durable subscriptions

Durable subscriptions continue to exist when a subscribing application's connection to the queue manager is closed. If a subscription is durable, when the subscribing application disconnects, the subscription remains in place and can be used by the subscribing application when it reconnects requesting the subscription again using the SubName that was returned when the subscription was created.

When subscribing durably, a subscription name (SubName) is required. Subscription names must be unique within a queue manager so that it can be used to identify a subscription. This means of identification is necessary when specifying a subscription you want to resume, if you have either deliberately closed the handle to the subscription (using the MQCO_KEEP_SUB option) or have been disconnected from the queue manager. You can resume an existing subscription by using the MQSUB call with the MQSO_RESUME option. Subscription names are also displayed if you use the DISPLAY SBSTATUS command with SUBTYPE ALL or ADMIN.

When an application no longer requires a durable subscription it can be removed using the MQCLOSE function call with the MQCO_REMOVE_SUB option or it can be deleted manually use the MQSC command DELETE SUB.

Whether durable subscriptions can be made to a topic can be controlled using the **DURSUB** topic attribute.

On return from an MQSUB call using the MQSO_RESUME option, subscription expiry is set to the original expiry of the subscription and not the remaining expiry time.

A queue manager continues to send publications to satisfy a durable subscription even if that subscriber application is not connected. This leads to a build up of messages on the subscriber queue. The easiest way to avoid this problem is to use a non-durable subscription wherever appropriate. However, where it is necessary to use durable subscriptions, a build up of messages can be avoided if the subscriber subscribes using the Retained publications option. A subscriber can then control when it receives publications by using the MQSUBRQ call.

Non-durable subscriptions

Non-durable subscriptions exist only as long as the subscribing application's connection to the queue manager remains open. The subscription is removed when the subscribing application disconnects from the queue manager either deliberately or by loss of connection. When the connection is closed, the information about the subscription is removed from the queue manager, and is no longer shown if you display subscriptions using the DISPLAY SBSTATUS command. No more messages are put to the subscriber queue.

What happens to any unconsumed publications on the subscriber queue for non-durable subscriptions is determined as follows.

- If a subscribing application is using a managed destination, any publications that have not been consumed are automatically removed.
- If the subscribing application provides a handle to its own subscriber queue when it subscribes, unconsumed messages are not removed automatically. It is the responsibility of the application to clear the queue if that is appropriate. If the queue is shared by more than one subscriber, or other point-to-point applications, it might not be appropriate to clear the queue completely.

Although not required for non durable subscriptions, a subscription name if provided, is used by the queue manager. Subscription names must be unique within the queue manager so that it can be used to identify a subscription.

Selection strings:

A *selection string* is an expression that is applied to a publication to determine whether it matches a subscription. Selection strings can include wildcard characters.

When you subscribe, in addition to specifying a topic, you can specify a selection string to select publications according to their message properties.

Topics

A topic is the subject of the information that is published in a publish/subscribe message.

Messages in point-to-point systems are sent to a specific destination address. Messages in subject-based publish/subscribe systems are sent to subscribers based on the subject that describes the contents of the message. In content-based systems, messages are sent to subscribers based on the contents of the message itself.

The WebSphere MQ publish/subscribe system is a subject-based publish/subscribe system. A publisher creates a message, and publishes it with a topic string that best fits the subject of the publication. To receive publications, a subscriber creates a subscription with a pattern matching topic string to select publication topics. The queue manager delivers publications to subscribers that have subscriptions that match the publication topic, and are authorized to receive the publications. The article, "Topic strings" on page 15, describes the syntax of topic strings that identify the subject of a publication. Subscribers also create topic strings to select which topics to receive. The topic strings that subscribers create can contain either of two alternative wildcard schemes to pattern match against the topic strings in publications. Pattern matching is described in "Wildcard schemes" on page 16.

In subject-based publish/subscribe, publishers, or administrators, are responsible for classifying subjects into topics. Typically subjects are organized hierarchically, into topic trees, using the '/' character to create subtopics in the topic string. See "Topic trees" on page 22 for examples of topic trees. Topics are nodes in the topic tree. Topics can be leaf-nodes with no further subtopics, or intermediate nodes with subtopics.

In parallel with organizing subjects into a hierarchical topic tree, you can associate topics with administrative topic objects. You assign attributes to a topic, such as whether the topic is distributed in a cluster, by associating it with an administrative topic object. The association is made by naming the topic using the TOPICSTR attribute of the administrative topic object. If you do not explicitly associate an administrative topic object to a topic, the topic inherits the attributes of its closest ancestor in the topic tree that you *have* associated with an administrative topic object. If you have not defined any parent topics at all, it inherits from SYSTEM.BASE.TOPIC. Administrative topic objects are described in "Administrative topic objects" on page 25.

Note: Even if you inherit all the attributes of a topic from SYSTEM.BASE.TOPIC, define a root topic for your topics that directly inherits from SYSTEM.BASE.TOPIC. For example, in the topic space of US states, USA/Alabama USA/Alaska, and so on, USA is the root topic. The main purpose of the root topic is to create discrete, non-overlapping topic spaces to avoid publications matching the wrong subscriptions. It also means you can change the attributes of your root topic to affect your whole topic space. For example, you might set the name for the **CLUSTER** attribute.

When you refer to a topic as a publisher or subscriber, you have a choice of supplying a topic string, referring to a topic object or you can do both, in which case the topic string you supply defines a subtopic of the topic object. The queue manager identifies the topic by appending the topic string to the topic string prefix named in the topic object, inserting an additional '/' in between the two topic strings, for example, *topic string/object string*. “Combining topic strings” on page 20 describes this further. The resulting topic string is used to identify the topic and associate it with an administrative topic object. The administrative topic object is not necessarily the same topic object as the topic object corresponding to the master topic.

In content based publish/subscribe, you define what messages you want to receive by providing selection strings that search the contents of every message. WebSphere MQ provides an intermediate form of content based publish/subscribe using message selectors that scan message properties rather than the full content of the message, see Selectors. The archetypal use of message selectors is to subscribe to a topic and then qualify the selection with a numeric property. The selector enables you to specify you are interested in values only in a certain range; something you cannot do using either character or topic-based wildcards. If you do need to filter based on the full content of the message, you need to use WebSphere Message Broker.

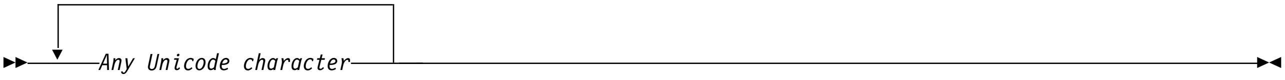
Topic strings:

Label information you publish as a topic using a topic string. Subscribe to groups of topics using either character or topic based wildcard topic strings.

Topics

A *topic string* is a character string that identifies the topic of a publish/subscribe message. You can use any characters you like when you construct a topic string.

Topic string



Three characters have special meaning in version 7 publish/subscribe. They are allowed anywhere in a topic string, but use them with caution. The use of the special characters is explained in “Topic-based wildcard scheme” on page 16.

A forward slash (/)

The topic level separator. Use the `'/'` character to structure the topic into a topic tree.

Avoid empty topic levels, `'//'`, if you can. These correspond to nodes in the topic hierarchy with no topic string. A leading or trailing `'/'` in a topic string corresponds to a leading or trailing empty node and should be avoided too.

The hash sign (#)

Used in combination with `'/'` to construct a multilevel wildcard in subscriptions. Take care using `'#'` adjacent to `'/'` in topic strings used to name published topics. “Examples of topic strings” on page 16 shows a sensible use of `'#'`.

The strings `'.../#/...'`, `'#/...'` and `'.../#'` have a special meaning in subscription topic strings. The strings match all topics at one or more levels in the topic hierarchy. Thus if you created a topic with one of those sequences, you could not subscribe to it, without also subscribing to all topics at multiple levels in the topic hierarchy.

The plus sign (+)

Used in combination with `'/'` to construct a single-level wildcard in subscriptions. Take care using `'+'` adjacent to `'/'` in topic strings used to name published topics.

The strings `'.../+/...'`, `'+/...'` and `'.../+'` have a special meaning in subscription topic strings. The strings match all topics at one level in the topic hierarchy. Thus if you created a topic with one of those sequences, you could not subscribe to it, without also subscribing to all topics at one level in the topic hierarchy.

Examples of topic strings

```
IBM/Business Area#/Results
IBM/Diversity/%African American
```

Wildcard schemes:

There are two wildcard schemes used to subscribe to multiple topics. The choice of scheme is a subscription option.

MQSO_WILDCARD_TOPIC
Select topics to subscribe to using the topic-based wildcard scheme.

This is the default if no wildcard schema is explicitly selected.

MQSO_WILDCARD_CHAR
Select topics to subscribe to using the character-based wildcard scheme.

Set either scheme by specifying the **wschema** parameter on the DEFINE SUB command. For more information, see DEFINE SUB.

Note: Subscriptions that were created before WebSphere MQ Version 7.0 always use the character-based wildcard scheme.

Examples

```
IBM/+/Results
#/Results
IBM/Software/Results
IBM/*ware/Results
```

Topic-based wildcard scheme:

Topic-based wildcards allow subscribers to subscribe to more than one topic at a time.

Topic-based wildcards are a powerful feature of the topic system in WebSphere MQ publish/subscribe. The multilevel wildcard and single level wildcard can be used for subscriptions, but they cannot be used within a topic by the publisher of a message.

The topic-based wildcard scheme allows you to select publications grouped by topic level. You can choose for *each level in the topic hierarchy*, whether the string in the subscription for that topic level must exactly match the string in the publication or not. For example the subscription, `IBM/+/Results` selects all the topics,

```
IBM/Software/Results
IBM/Services/Results
IBM/Hardware/Results
```

There are two types of wildcard.

Multilevel wildcard

- The multilevel wildcard is used in subscriptions. When used in a publication it is treated as a literal.
- The multilevel wildcard character `'#'` is used to match any number of levels within a topic. For example, using the example topic tree, if you subscribe to `'USA/Alaska/#'`, you receive messages on topics `'USA/Alaska'` and `'USA/Alaska/Juneau'`.

- The multilevel wildcard can represent zero or more levels. Therefore, 'USA/#' can also match the singular 'USA', where '#' represents zero levels. The topic level separator is meaningless in this context, because there are no levels to separate.
- The multilevel wildcard is only effective when specified on its own or next to the topic level separator character. Therefore, '#' and 'USA/#' are valid topics where the '#' character is treated as a wildcard. However, although 'USA#' is also a valid topic string, the '#' character is not regarded as a wildcard and does not have any special meaning. See "When topic-based wildcards are not wild" on page 19 for more information.

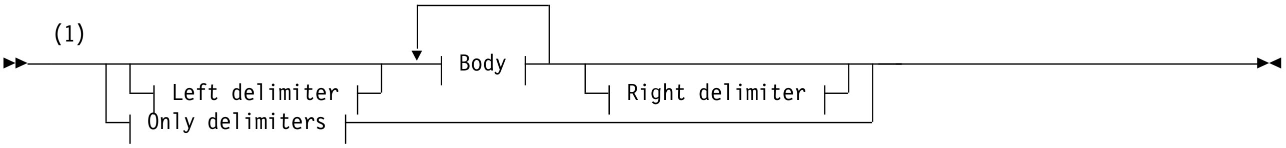
Single level wildcard

- The single wildcard is used in subscriptions. When used in a publication it is treated as a literal.
- The single-level wildcard character '+' matches one, and only one, topic level. For example, 'USA/+' matches 'USA/Alabama', but not 'USA/Alabama/Auburn'. Because the single-level wildcard matches only a single level, 'USA/+' does not match 'USA'.
- The single-level wildcard can be used at any level in the topic tree, and in conjunction with the multilevel wildcard. The single-level wildcard must be specified next to the topic level separator, except when it is specified on its own. Therefore, '+' and 'USA/+' are valid topics where the '+' character is treated as a wildcard. However, although 'USA+' is also a valid topic string, the '+' character is not regarded as a wildcard and does not have any special meaning. See "When topic-based wildcards are not wild" on page 19 for more information.

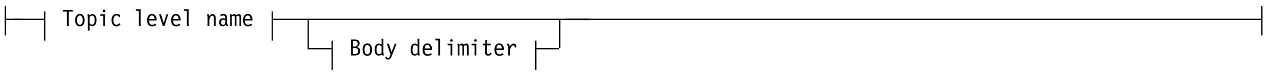
The syntax for the topic-based wildcard scheme has no escape characters. Whether '#' and '+' are treated as wildcards or not depends on their context. See "When topic-based wildcards are not wild" on page 19 for more information.

Note: The beginning and end of a topic string is treated in a special way. Using '$' to denote the end of the string, then '$#/...' is a multilevel wildcard, and '$/#/..'. is an empty node at the root, followed by a multilevel wildcard.

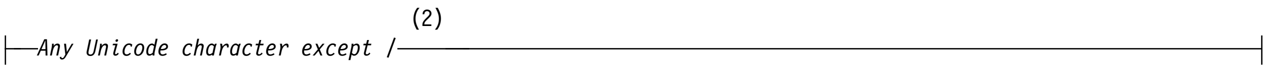
Topic-based wildcard string



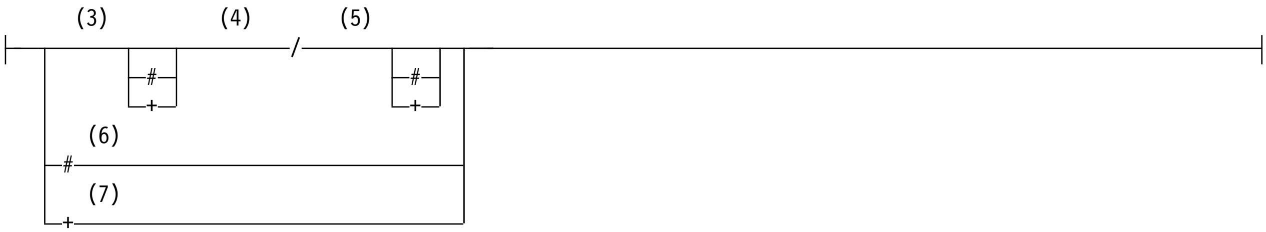
Body:



Topic level name:



Only delimiters:



Notes:

1 A null or zero length topic string is invalid

2 You are advised not to use any of `*, ?, %` in level name strings for compatibility between char based and topic-based wildcard schemes.

3 These cases are equivalent to the *left delimiter* pattern.

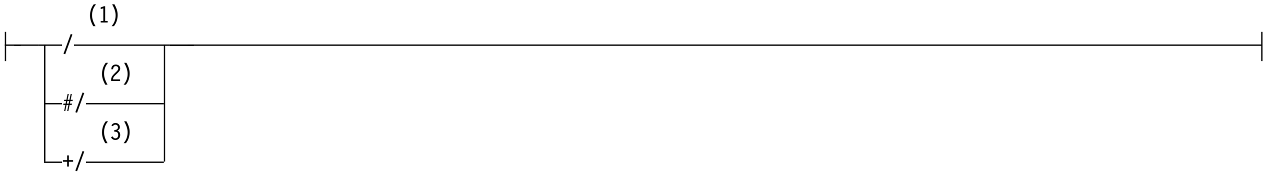
4 / with no wildcards matches a single empty topic.

5 These cases are equivalent to the *right delimiter* pattern.

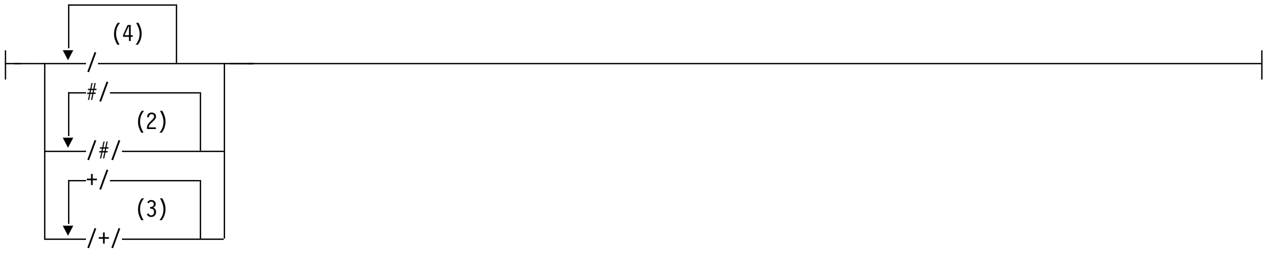
6 Match every topic.

7 Match every topic where there is only one level.

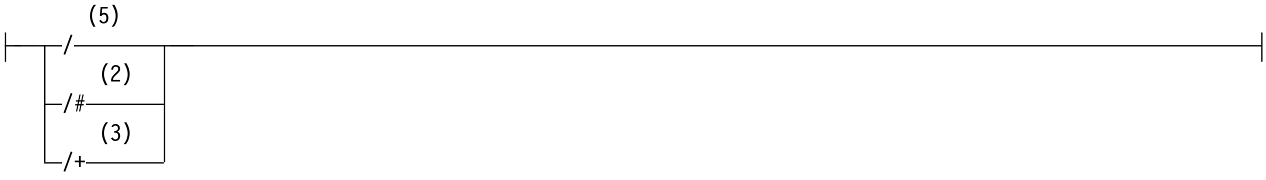
Left delimiter:



Body delimiter:



Right delimiter:



Notes:

1 The topic string starts with an empty topic

2 Matches zero or more levels. Multiple multi-level match strings have the same effect as one multi-level match string.

3 Matches exactly one level.

4 // is an empty topic - a topic object with no topic string.

5 The topic string ends with an empty topic

When topic-based wildcards are not wild

The wildcard characters '+' and '#' have no special meaning when they are mixed with other characters (including themselves) in a topic level.

This means that topics that contain '+' or '#' together with other characters in a topic level can be published.

For example, consider the following two topics:

1. level0/level1/+/level4/#
2. level0/level1/#+/level4/level#

In the first example, the characters '+' and '#' are treated as wildcards and are therefore not valid in a topic string that is to be published to but are valid in a subscription.

In the second example, the characters '+' and '#' are not treated as wildcards and therefore the topic string can be both published and subscribed to.

Examples

```
IBM/+/Results
#/Results
IBM/Software/Results
```

Character-based wildcard scheme:

The character-based wildcard scheme allows you to select topics based on traditional character matching.

You can select all topics at multiple levels in a topic hierarchy using the string '*'. Using '*' in the character-based wildcard scheme is equivalent to using the topic-based wildcard string '#'

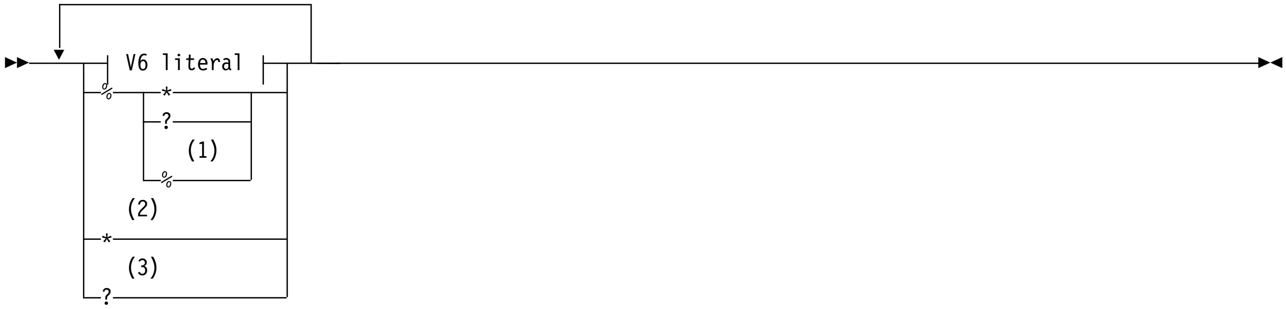
'*x*/*/*y*' is equivalent to '*x*/#/*y*' in the topic-based scheme, and selects all topics in the topic hierarchy between levels '*x* and *y*', where '*x*' and '*y*' are topic names that are not in the set of levels returned by the wildcard.

'/+/' in the topic-based scheme has no exact equivalent in the character-based scheme. 'IBM/*/Results' would also select 'IBM/Patents/Software/Results'. Only if the set of topic names at each level of the hierarchy are unique, can you always construct queries with the two schemes that yield identical matches.

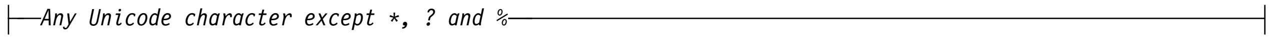
Used in a general way, '*' and '?' in the character-based scheme have no equivalents in the topic-based scheme. The topic-based scheme does not perform partial matching using wildcards. The character based wildcard subscription 'IBM/*ware/Results' has no topic-based equivalent.

Note: Matches using character wildcard subscriptions are slower than matches using topic-based subscriptions.

Character-based wildcard string



V6 literal:



Notes:

1 Means "Escape the following character", so that it is treated as a literal. `'%'` must be followed by either `'*'`, `'?'` or `'%'`. See “Examples of topic strings” on page 16.

2 Means "Match zero or more characters" in a subscription.

3 Means "Match exactly one character" in a subscription.

Examples

```
IBM/*/Results
IBM/*ware/Results
```

Combining topic strings:

When creating subscriptions, or opening topics so you can publish messages to them, the topic string can be formed by combining two separate sub-topic strings, or “subtopics”. One subtopic is provided by the application or administrative command as a topic string, and the other is the topic string associated with a topic object. You can use either subtopic as the topic string on its own, or combine them to form a new topic name.

For example, when you define a subscription using the MQSC command **DEFINE SUB**, the command can take either **TOPICSTR** (topic string) or **TOPICOBJ** (topic object) as an attribute, or both together. If only **TOPICOBJ** is provided, the topic string associated with that topic object is used as the topic string. If only **TOPICSTR** is provided, that is used as the topic string. If both are provided, they are concatenated to form a single topic string in the form **TOPICOBJ/TOPICSTR**, where the **TOPICOBJ** configured topic string is always first and the two parts of the string are always separated by a “/” character.

Similarly, in an MQI program the full topic name is created by MQOPEN. It is composed of two fields used in publish/subscribe MQI calls, in the order listed:

1. The **TOPICSTR** attribute of the topic object, named in the **ObjectName** field.
2. The **ObjectString** parameter defining the subtopic provided by the application.

The resulting topic string is returned in the **ResObjectString** parameter.

These fields are considered to be present if the first character of each field is not a blank or null character, and the field length is greater than zero. If only one of the fields is present, it is used unchanged as the topic name. If neither field has a value, the call fails with reason code MQRC_UNKNOWN_OBJECT_NAME, or MQRC_TOPIC_STRING_ERROR if the full topic name is not valid.

If both fields are present, a "/" character is inserted between the two elements of the resultant combined topic name.

Table 2 shows examples of topic string concatenation:

Table 2. Topic string concatenation examples

TOPICSTR of the topic object	**Topic string provided by application or DEFINE SUB command**	**Full topic name**	**Comment**
Football/Scores	''	Football/Scores	The TOPICSTR of the topic object is used alone.
''	Football/Scores	Football/Scores	The ObjectString/ TOPICSTR is used alone.
Football	Scores	Football/Scores	A "/" character is added at the concatenation point.
Football	/Scores	Football//Scores	An 'empty node' is produced between the two strings. This is different to "Football/Scores".
/Football	Scores	/Football/Scores	The topic starts with an 'empty node'. This is different to "Football/Scores".

The "/" character is considered as a special character, providing structure to the full topic name in "Topic trees" on page 22. The "/" character must not be used for any other reason, because the structure of the topic tree is affected. The topic "/Football" is not the same as the topic "Football".

Note: If you use a topic object when creating a subscription, the value of the topic object topic string is fixed in the subscription at define time. Any subsequent change to the topic object does not affect the topic string that the subscription is defined to.

Wildcard characters in topic strings

The following wildcard characters are special characters:
- plus sign (+)
- number sign (#)
- asterisk (*)
- question mark (?)

Wildcard characters only have special meaning when used by a subscription. These characters are not considered as invalid when used elsewhere, however you must ensure you understand how they are used and you might prefer not to use these characters in your topic strings when publishing or defining topic objects.

If you publish on a topic string with # or + mixed in with other characters (including themselves) within a topic level, the topic string can be subscribed to with either wildcard scheme.

If you publish on a topic string with # or + as the only character between two / characters, the topic string cannot be subscribed to explicitly by an application using the wildcard scheme MQSO_WILDCARD_TOPIC. This situation results in the application getting more publications than expected.

You should not use a wildcard character in the topic string of a defined topic object. If you do this, the character is treated as a literal character when the object is used by a publisher, and as a wildcard character when used by a subscription. This can lead to confusion.

Example code snippet

This code snippet, extracted from the example program Example 2: Publisher to a variable topic, combines a topic object with a variable topic string:

```
MQOD     td = {MQOD_DEFAULT};  /* Object Descriptor           */
td.ObjectType = MQOT_TOPIC;    /* Object is a topic           */
td.Version = MQOD_VERSION_4;   /* Descriptor needs to be V4    */
strncpy(td.ObjectName, topicName,  MQ_TOPIC_NAME_LENGTH);
td.ObjectString.VSPtr = topicString;
td.ObjectString.VSLength = (MQLONG)strlen(topicString);
td.ResObjectString.VSPtr = resTopicStr;
td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
```

Topic trees:

Each topic that you define is an element, or node, in the topic tree. The topic tree can either be empty to start with or contain topics that have been defined previously using MQSC or PCF commands. You can define a new topic either by using the create topic commands or by specifying the topic for the first time in a publication or subscription.

Although you can use any character string to define a topic's topic string, it is advisable to choose a topic string that fits into a hierarchical tree structure. Thoughtful design of topic stings and topic trees can help you with the following operations:

- Subscribing to multiple topics.
- Establishing security policies.

Although you can construct a topic tree as a flat, linear structure, it is better to build a topic tree in a hierarchical structure with one or more root topics. For more information about security planning and topics, see Publish/subscribe security.

Figure 5 shows an example of a topic tree with one root topic.

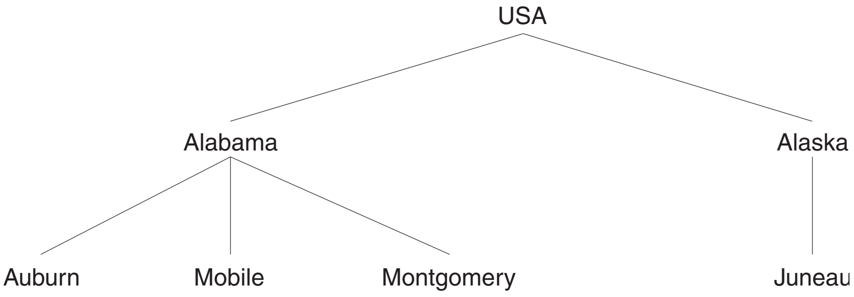


Figure 5. Example of a topic tree

Each character string in the figure represents a node in the topic tree. A complete topic string is created by aggregating nodes from one or more levels in the topic tree. Levels are separated by the "/" character. The format of a fully specified topic string is: "root/level2/level3".

The valid topics in the topic tree shown in Figure 5 on page 22 are:

"USA"
"USA/Alabama"
"USA/Alaska"
"USA/Alabama/Auburn"
"USA/Alabama/Mobile"
"USA/Alabama/Montgomery"
"USA/Alaska/Juneau"

When you design topic strings and topic trees, remember that the queue manager does not interpret, or attempt to derive meaning from, the topic string itself. It simply uses the topic string to send selected messages to subscribers of that topic.

The following principles apply to the construction and content of a topic tree:

- There is no limit to the number of levels in a topic tree.
- There is no limit to the length of the name of a level in a topic tree.
- There can be any number of "root" nodes; that is, there can be any number of topic trees.

Reducing the number of unwanted topics in the topic tree:

The performance of a publish/subscribe system is improved by reducing the number of unwanted topics in the topic tree. What is an unwanted topic and how do you remove them?

You can create large numbers of topics without affecting performance adversely. However, some ways of using publish/subscribe result in continually expanding topic trees. An exceptionally large number of topics are created once and never used again. The growing number of topics might become a performance problem.

How can you avoid designs that lead to a large and growing number of unwanted topics? What can you do to help the queue manager remove unwanted topics from the topic tree?

The queue manager recognizes an unwanted topic because it has been unused for 30 minutes. The queue manager removes unused topics from the topic tree for you. The 30 minute duration can be changed by altering the queue manager attribute, **TREELIFE**. You can help the queue manager to remove unwanted topics by making sure that the topic appears to the queue manager to be unused. The section, “What is an unused topic?” explains what an unused topic is.

A programmer, designing any application, and especially designing a long running application, considers its resource usage: how much resource the program requires, are there any unbounded demands, and any resource leaks? Topics are a resource that publish/subscribe programs use. Scrutinize the use of topics just like any other resource a program uses.

What is an unused topic?

Before defining what an unused topic is, what exactly counts as a topic?

When a topic string, such as `USA/Alabama/Auburn` , is converted into a topic, the topic is added to the topic tree. Additional topic nodes, and their corresponding topics, are created in the tree, if necessary. The topic string `USA/Alabama/Auburn` is converted into a tree with three topics.

- `USA`
- `USA/Alabama`
- `USA/Alabama/Auburn`

To display all the topics in the topic tree, use the **runmqsc** command `DISPLAY TPSTATUS('#') TYPE(TOPIC)` .

An unused topic in the topic tree has the following properties.

It is not associated with a topic object

An administrative topic object has a topic string that associates it with a topic. When you define the topic object `Alabama` , if the topic, `USA/Alabama`, it is to be associated with does not exist, the topic is created from the topic string. If the topic does exist, the topic object and the topic are associated together using the topic string.

It does not have a retained publication

A topic with a retained publication results from a publisher putting a message to a topic with the option `MQPMO_RETAIN` .

Use the **runmqsc** command `DISPLAY TPSTATUS('USA/Alabama') RETAINED` to check if `USA/Alabama` has a retained publication. The response is `YES` or `NO`.

Use the **runmqsc** command `CLEAR TOPICSTR('USA/Alabama') CLTRTYPE(RETAINED)` to remove a retained publication from `USA/Alabama` .

It has no child topics

`USA/Alabama/Auburn` is a topic with no child topics. `USA/Alabama/Auburn` is the direct child topic of `USA/Alabama`.

Display the direct children of `USA/Alabama` with the **runmqsc** command `DISPLAY TPSTATUS('USA/Alabama/+')` .

There are no active publishers to the node

An active publisher to a node is an application that has the topic open for output.

For example, an application opens the topic object named **Alabama** with open options `MQOO_OUTPUT`.

To display active publishers to `USA/Alabama` and all its children, use the **runmqsc** command `DISPLAY TPSTATUS('USA/Alabama/#') TYPE(PUB) ACTCONN`.

There are no active subscribers to the node

An active subscriber can either be a durable subscription, or an application that has registered a subscription to a topic with MQSUB, and not closed it.

To display active subscriptions to `USA/Alabama`, use the **runmqsc** command `DISPLAY TPSTATUS('USA/Alabama') TYPE(SUB) ACTCONN`.

To display active subscriptions to `USA/Alabama` and all its children, use the **runmqsc** command `DISPLAY TPSTATUS('USA/Alabama/#') TYPE(SUB) ACTCONN`.

Reducing the number of topics in a topic tree

In summary, there are a number of ways to reduce the number of topics in a topic tree.

Modify `TREELIFE`

An unused topic has a lifetime of 30 minutes by default. You can make the lifetime of an unused topic smaller.

For example, The **runmqsc** command, `ALTER QMGR TREELIFE(900)`, reduces lifetime of an unused topic from 30 minutes to 15 minutes.

Exceptionally, restart the queue manager

When the queue manager is restarted, the topic tree is reinitialized from topic objects, nodes with retained publications, and durable subscriptions. Topics that had been created by the operation of publisher and subscriber programs are eliminated.

Use the **runmqsc** command `DISPLAY TPSTATUS('#') TYPE(TOPIC)` periodically to list all topics and check if the number is growing.

As a last resort, if the growth in unwanted topics has been the cause of performance problems in the past, restart the queue manager.

Administrative topic objects:

Using an administrative topic object, you can assign specific, non-default attributes to topics.

Figure 6 shows how a high-level topic of `Sport` divided into separate topics covering different sports can be visualized as a topic tree:

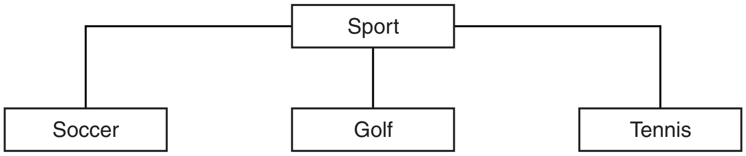


Figure 6. Visualization of a topic tree

Figure 7 shows how the topic tree can be divided further, to separate different types of information about each sport:

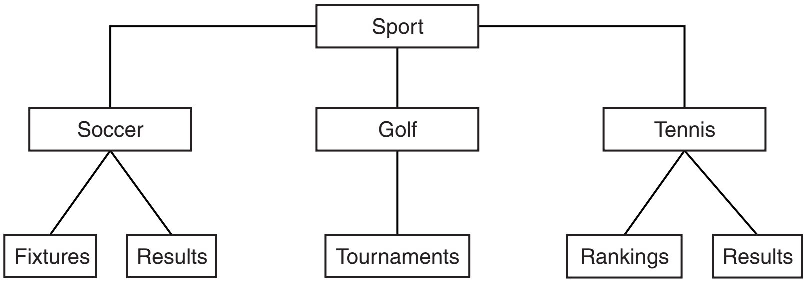


Figure 7. Extended topic tree

To create the topic tree illustrated, no administrative topic objects need be defined. Each of the nodes in this tree are defined by a topic string created in a publish or subscribe operation. Each topic in the tree inherits its attributes from its parent. Attributes are inherited from the parent topic object, because by default all attributes are set to `ASPARENT`. In this example, every topic has the same attributes as the `Sport` topic. The `Sport` topic has no administrative topic object, and inherits its attributes from SYSTEM.BASE.TOPIC.

Note, that it is not good practice to give authorities for non-mqm users at the root node of the topic tree, which is SYSTEM.BASE.TOPIC, because the authorities are inherited but cannot be restricted. Therefore, by giving authorities at this level, you are giving authorities to the whole tree. You should give the authority at a lower topic level in the hierarchy.

Administrative topic objects can be used to define specific attributes for particular nodes in the topic tree. In the following example, the administrative topic object is defined to set the durable subscriptions property `DURSUB`of the soccer topic to the value `NO`:

```
DEFINE TOPIC(FOOTBALL.EUROPEAN)
       TOPICSTR('Sport/Soccer')
       DURSUB(NO)
       DESCR('Administrative topic object to disallow durable subscriptions')
```

The topic tree can now be visualized as:

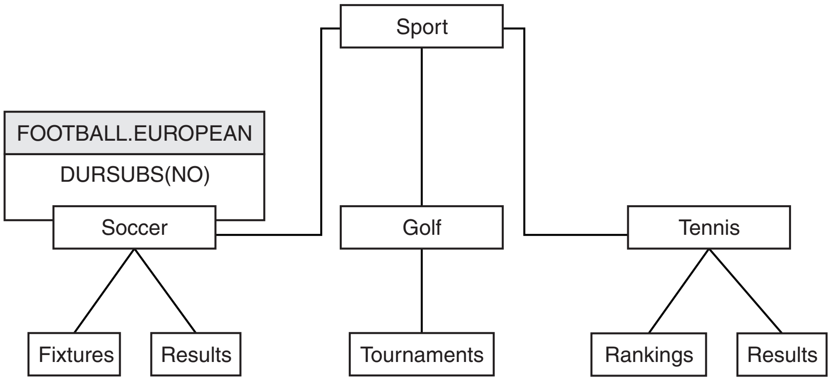


Figure 8. Visualization of an administrative topic object associated with the Sport/Soccer topic

Any applications subscribing to topics beneath Soccer in the tree can still use the topic strings they used before the administrative topic object was added. However, an application can now be written to subscribe using the object name `FOOTBALL.EUROPEAN`, instead of the string `/Sport/Soccer`. For example, to subscribe to `/Sport/Soccer/Results`, an application can specify `MQSD.ObjectName` as `FOOTBALL.EUROPEAN` and `MQSD.ObjectString` as `Results` .

With this feature, you can hide part of the topic tree from application developers. Define an administrative topic object at a particular node in the topic tree, then application developers can define their own topics as children of the node. Developers must know about the parent topic, but not about any other nodes in the parent tree.

Inheriting attributes

If a topic tree has many administrative topic objects, each administrative topic object, by default, inherits its attributes from its closest parent administrative topic. The previous example has been extended in Figure 9:

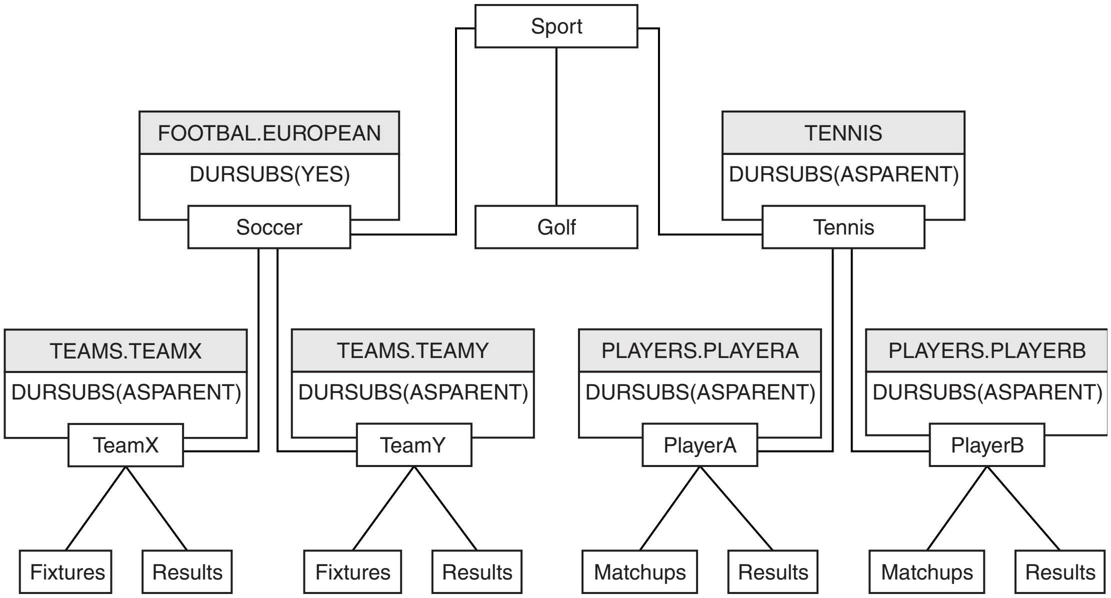


Figure 9. Topic tree with several administrative topic objects

For example use inheritance to give all the child topics of /Sport/Soccer the property that subscriptions are non-durable. Change the DURSUB attribute of FOOTBALL.EUROPEAN to NO.

This attribute can be set using the following command:

```
ALTER TOPIC(FOOTBALL.EUROPEAN) DURSUB(NO)
```

All the administrative topic objects of child topics of Sport/Soccer have the property DURSUB set to the default value ASPARENT. After changing the DURSUB property value of FOOTBALL.EUROPEAN to NO, the child topics of Sport/Soccer inherit the DURSUB property value NO . All child topics of Sport/Tennis inherit the value of DURSUB from SYSTEM.BASE.TOPIC object. SYSTEM.BASE.TOPIC has the value of YES.

Trying to make a durable subscription to the topic Sport/Soccer/TeamX/Results would now fail; however, trying to make a durable subscription to Sport/Tennis/PlayerB/Results would succeed.

Controlling wildcard usage with the WILDCARD property

Use the MQSC **Topic** WILDCARD property or the equivalent PCF Topic WildcardOperation property to control the delivery of publications to subscriber applications that use wildcard topic string names. The WILDCARD property can have one of two possible values:

WILDCARD

The behavior of wildcard subscriptions with respect to this topic.

PASSTHRU

Subscriptions made to a wildcarded topic less specific than the topic string at this topic object receive publications made to this topic and to topic strings more specific than this topic.

BLOCK

Subscriptions made to a wildcarded topic less specific than the topic string at this topic object do not receive publications made to this topic or to topic strings more specific than this topic.

The value of this attribute is used when subscriptions are defined. If you alter this attribute, the set of topics covered by existing subscriptions is not affected by the modification. This scenario applies also if the topology is changed when topic objects are created or deleted; the set of topics matching subscriptions created following the modification of the WILDCARD attribute is created using the modified topology. If you want to force the matching set of topics to be re-evaluated for existing subscriptions, you must restart the queue manager.

In the example, “Example: Create the Sport publish/subscribe cluster” on page 32, you can follow the steps to create the topic tree structure shown in Figure 10 on page 28.

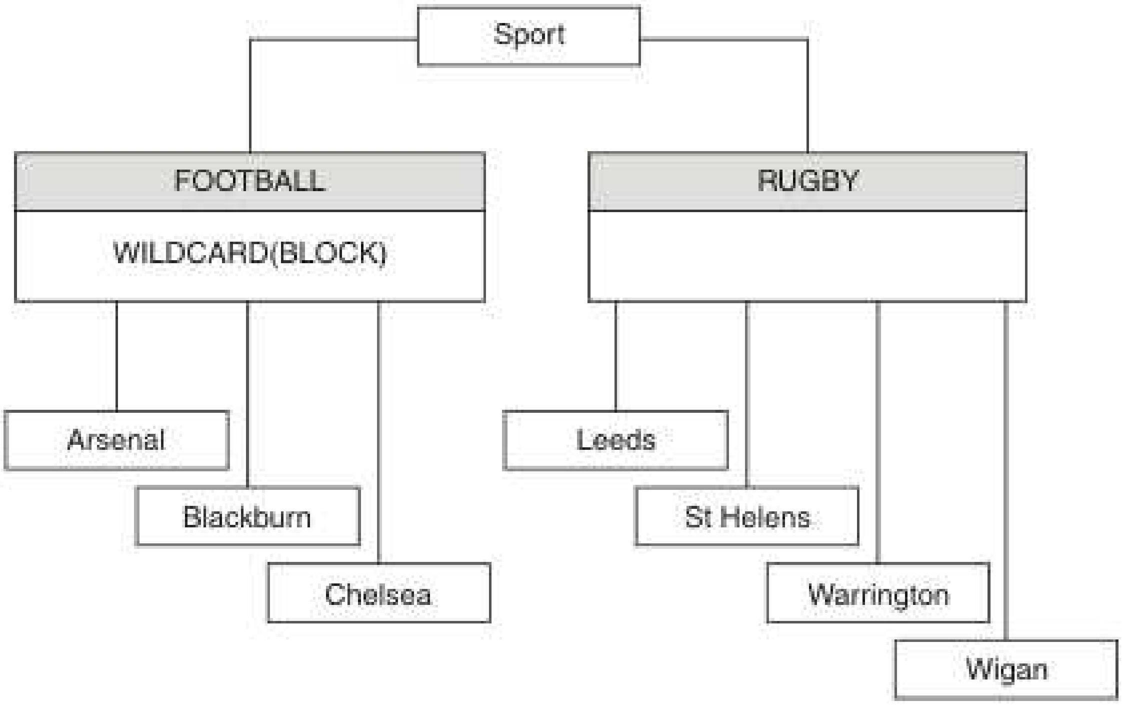


Figure 10. A topic tree that uses the WILDCARD property, `BLOCK`

A subscriber using the wildcard topic string # receives all publications to the Sport topic and the Sport/Rugby subtree. The subscriber receives no publications to the Sport/Football subtree, because the WILDCARD property value of the Sport/Football topic is BLOCK.

PASSTHRU is the default setting. You can set the WILDCARD property value PASSTHRU to nodes in the Sport tree. If the nodes do not have the WILDCARD property value BLOCK, setting PASSTHRU does not alter the behavior observed by subscribers to nodes in the Sports tree.

In the example, create subscriptions to see how the wildcard setting affects the publications that are delivered; see Figure 14 on page 33. Run the publish command in Figure 17 on page 34 to create some publications.

```
pub QMA
```

Figure 11. Publish to QMA

The results are shown in Table 3 on page 29. Notice how setting the WILDCARD property value BLOCK, prevents subscriptions with wildcards from receiving publications to topics within the scope of the wildcard.

Table 3. Publications received on QMA

Subscription	Topic string	Publications received	Notes®
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football prevents wildcard subscription on Arsenal
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Default WILDCARD on Sports/Rugby does not prevent wildcard subscription on Leeds.

Note:

Suppose a subscription has a wildcard that matches a topic object with the WILDCARD property value BLOCK. If the subscription also has a topic string to the right of the matching wildcard, the subscription never receives a publication. The set of publications that are not blocked are publications to topics that are parents of the blocked wildcard. Publications to topics that are children of the topic with the BLOCK property value are blocked by the wildcard. Therefore subscription topic strings that include a topic to the right of the wildcard never receive any publications to match.

Setting the WILDCARD property value to BLOCK does not mean you cannot subscribe using a topic string that includes wildcards. Such a subscription is normal. The subscription has an explicit topic that matches the topic with a topic object having a WILDCARD property value BLOCK. It uses wildcards for topics that are parents or children of the topic with the WILDCARD property value BLOCK. In the example in Figure 10 on page 28, a subscription such as Sports/Football/# can receive publications.

Wildcards and cluster topics

Cluster topic definitions are propagated to every queue manager in a cluster. A subscription to a cluster topic at one queue manager in a cluster results in the queue manager creating proxy subscriptions. A proxy subscription is created at every other queue manager in the cluster. Subscriptions using topics strings containing wildcards, combined with cluster topics, can give hard to predict behavior. The behavior is explained in the following example.

In the cluster set up for the example, "Example: Create the Sport publish/subscribe cluster" on page 32, QMB has the same set of subscriptions as QMA, yet QMB received no publications after the publisher published to QMA, see Figure 11 on page 28. Although the Sports/Football and Sports/Rugby topics are cluster topics, the subscriptions defined in fullsubs.tst do not reference a cluster topic. No proxy subscriptions are propagated from QMB to QMA. Without proxy subscriptions, no publications to QMA are forwarded to QMB .

Some of the subscriptions, such as Sports/#/Leeds, might seem to reference a cluster topic, in this case Sports/Rugby . The Sports/#/Leeds subscription actually resolves to the topic object SYSTEM.BASE.TOPIC.

The rule for resolving the topic object referenced by a subscription such as, Sports/#/Leeds is as follows. Truncate the topic string to the first wildcard. Scan left through the topic string looking for the first topic that has an associated administrative topic object. The topic object might specify a cluster name, or define a local topic object. In the example, Sports/#/Leeds, the topic string after truncation is Sports, which has no topic object, and so Sports/#/Leeds inherits from SYSTEM.BASE.TOPIC, which is a local topic object.

To see how subscribing to clustered topics can change the way wildcard propagation works, run the batch script, upsubs.bat. The script clears the subscription queues, and adds the cluster topic subscriptions in fullsubs.tst. Run puba.bat again to create a batch of publications; see Figure 11 on page 28.

Table 4 shows the result of adding two new subscriptions to the same queue manager that the publications were published on. The result is as expected, the new subscriptions receive one publication each, and the numbers of publications received by the other subscriptions are unchanged. The unexpected results occur on the other cluster queue manager; see Table 5.

Table 4. Publications received on QMA

Subscription	Topic string	Publications received	Notes
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football prevents wildcard subscription on Arsenal
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Default WILDCARD on Sports/Rugby does not prevent wildcard subscription on Leeds.
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal receives a publication because the subscription does not have a wildcard.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds would receive a publication in any event.

Table 5 shows the results of adding the two new subscriptions on QMB and publishing on QMA. Recall that QMB received no publications without these two new subscriptions. As expected, the two new subscriptions receive publications, because Sports/FootBall and Sports/Rugby are both cluster topics. QMB forwarded proxy subscriptions for Sports/Football/Arsenal and Sports/Rugby/Leeds to QMA, which then sent the publications to QMB.

The unexpected result is that the two subscriptions Sports/# and Sports/#/Leeds that previously received no publications, now receive publications. The reason is that the Sports/Football/Arsenal and Sports/Rugby/Leeds publications forwarded to QMB for the other subscriptions are now available for any subscriber attached to QMB . Consequently the subscriptions to the local topics Sports/# and Sports/#/Leeds receive the Sports/Rugby/Leeds publication. Sports/#/Arsenal continues not to receive a publication, because Sports/Football has its WILDCARD property value set to BLOCK.

Table 5. Publications received on QMB

Subscription	Topic string	Publications received	Notes
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football*
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football prevents wildcard subscription on Arsenal
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Default WILDCARD on Sports/Rugby does not prevent wildcard subscription on Leeds .*
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal receives a publication because the subscription does not have a wildcard.

Table 5. Publications received on QMB (continued)

Subscription	Topic string	Publications received	Notes
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds would receive a publication in any event.

In most applications, it is undesirable for one subscription to influence the behavior of another subscription. One important use of the WILDCARD property with the value BLOCK is to make the subscriptions to the same topic string containing wildcards behave uniformly. Whether the subscription is on the same queue manager as the publisher, or a different one, the results of the subscription are the same.

Wildcards and streams

WebSphere MQ Version 6 streams are mapped to topics by WebSphere MQ Version 7; see "Streams and topics" on page 35 . In the default mapping, which is performed by **strmqbrk** in Version 7, all the topics in the stream Sports would be mapped to the topic Sports. All the topics in the stream Business would be mapped to the topic Business.

A subscription in WebSphere MQ Version 6 to * in the Sports stream receives all publications in the Sports tree, and no publications in the Business tree. The same subscription in version 7 would receive all the publications in the Sports tree and all the publications in the Business tree. To block this behavior, when streams are migrated to version 7, **strmqbrk** sets the WILDCARD property. It sets it to the value BLOCK for each of the top-level topics that are migrated from a stream. The WILDCARD property of Sports and Business is set to the value BLOCK by the conversion from the version 6 streams called Sports and Business.

For a new application written to the publish/subscribe API, the effect is that a subscription to * receives no publications. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

The behavior of an existing queued publish/subscribe application does not change when the publish/subscribe broker is migrated to WebSphere MQ Version 7. The **StreamName** property in the **Publish**, **Register Publisher**, or **Subscriber** commands is mapped to the name of the topic the stream has been migrated to.

Wildcards and subscription points

WebSphere Message Broker subscriptions points are mapped to topics by WebSphere MQ Version 7; see "Subscription points and topics" on page 37. In the default mapping, which is performed by **migmqbrk** in Version 7, all the topics in the subscription point Sports would be mapped to the topic Sports. All the topics in the subscription point Business would be mapped to the topic Business.

A subscription on WebSphere Message Broker Version 6 to * in the Sports subscription point receives all publications in the Sports tree, and no publications in the Business tree. The same subscription in version 7 would receive all the publications in the Sports tree and all the publications in the Business tree. To block this behavior, when subscription points are migrated to version 7, **migmqbrk** sets the WILDCARD property. It sets it to the value BLOCK for each of the top-level topics that are migrated from a subscription point. The WILDCARD property of Sports and Business is set to the value BLOCK by the conversion from the WebSphere Message Broker subscription points called Sports and Business .

For a new application written to the publish/subscribe API, the effect of the migration is that a subscription to * receives no publications. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

The behavior of an existing queued publish/subscribe application does not change when the publish/subscribe broker is migrated to WebSphere MQ Version 7. The **SubPoint** property in the **Publish**, **Register Publisher**, or **Subscriber** commands is mapped to the name of the topic the subscription has been migrated to.

Example: Create the Sport publish/subscribe cluster

The steps that follow create a cluster, CL1, with four queue managers: two full repositories, CL1A and CL1B, and two partial repositories, QMA and QMB. The full repositories are used to hold only cluster definitions. QMA is designated the cluster topic host. Durable subscriptions are defined on both QMA and QMB.

Note: The example is coded for Windows. You must recode Create qmgrs.bat and create pub.bat to configure and test the example on other platforms.

1. Create the script files.
   a. Create topics.tst
   b. Create wildsubs.tst
   c. Create fullsubs.tst
   d. Create qmgrs.bat
   e. create pub.bat
2. Run Create qmgrs.bat to create the configuration.

   ```
   qmgrs
   ```

Create the topics in Figure 10 on page 28. The script in figure 5 creates the cluster topics Sports/Football and Sports/Rugby.

Note: The REPLACE option does not replace the TOPICSTR properties of a topic. TOPICSTR is a property that is usefully varied in the example to test different topic trees. To change topics, delete the topic first.

```
DELETE TOPIC ('Sports')
DELETE TOPIC ('Football')
DELETE TOPIC ('Arsenal')
DELETE TOPIC ('Blackburn')
DELETE TOPIC ('Chelsea')
DELETE TOPIC ('Rugby')
DELETE TOPIC ('Leeds')
DELETE TOPIC ('Wigan')
DELETE TOPIC ('Warrington')
DELETE TOPIC ('St. Helens')

DEFINE TOPIC ('Sports')     TOPICSTR('Sports')
DEFINE TOPIC ('Football')   TOPICSTR('Sports/Football') CLUSTER(CL1) WILDCARD(BLOCK)
DEFINE TOPIC ('Arsenal')    TOPICSTR('Sports/Football/Arsenal')
DEFINE TOPIC ('Blackburn')  TOPICSTR('Sports/Football/Blackburn')
DEFINE TOPIC ('Chelsea')    TOPICSTR('Sports/Football/Chelsea')
DEFINE TOPIC ('Rugby')      TOPICSTR('Sports/Rugby')    CLUSTER(CL1)
DEFINE TOPIC ('Leeds')      TOPICSTR('Sports/Rugby/Leeds')
DEFINE TOPIC ('Wigan')      TOPICSTR('Sports/Rugby/Wigan')
DEFINE TOPIC ('Warrington') TOPICSTR('Sports/Rugby/Warrington')
DEFINE TOPIC ('St. Helens') TOPICSTR('Sports/Rugby/St. Helens')
```

Figure 12. Delete and create topics: topics.tst

Note: Delete the topics, as REPLACE does not replace topic strings.

Create subscriptions with wildcards. The wildcards corresponding the topics with topic objects in Figure 10 on page 28. Create a queue for each subscription. The queues are cleared and the subscriptions deleted when the script is run or rerun.

Note: The REPLACE option does not replace TOPICOBJ or TOPICSTR properties of a subscription. TOPICOBJ or TOPICSTR are the properties that are usefully varied in the example to test different subscriptions. To change them, delete the subscription first.

```
DEFINE QLOCAL(QSPORTS)   REPLACE
DEFINE QLOCAL(QSARSENAL) REPLACE
DEFINE QLOCAL(QSLEEDS)   REPLACE
CLEAR  QLOCAL(QSPORTS)
CLEAR  QLOCAL(QSARSENAL)
CLEAR  QLOCAL(QSLEEDS)

DELETE SUB   (SPORTS)
DELETE SUB   (SARSENAL)
DELETE SUB   (SLEEDS)
DEFINE SUB   (SPORTS)   TOPICSTR('Sports/#')         DEST(QSPORTS)
DEFINE SUB   (SARSENAL) TOPICSTR('Sports/+/Arsenal') DEST(QSARSENAL)
DEFINE SUB   (SLEEDS)   TOPICSTR('Sports/+/Leeds')   DEST(QSLEEDS)
```

Figure 13. Create wildcard subscriptions: wildsubs.tst

Create subscriptions that reference the cluster topic objects.

Note:

The delimiter, /, is automatically inserted between the topic string referenced by TOPICOBJ, and the topic string defined by TOPICSTR.

The definition, DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) creates the same subscription. TOPICOBJ is used as a quick way to reference topic string you have already defined. The subscription, when created, no longer refers to the topic object.

```
DEFINE QLOCAL(QFARSENAL) REPLACE
DEFINE QLOCAL(QRLEEDS)   REPLACE
CLEAR  QLOCAL(QFARSENAL)
CLEAR  QLOCAL(QRLEEDS)

DELETE SUB   (FARSENAL)
DELETE SUB   (RLEEDS)
DEFINE SUB   (FARSENAL) TOPICOBJ('Football') TOPICSTR('Arsenal') DEST(QFARSENAL)
DEFINE SUB   (RLEEDS)   TOPICOBJ('Rugby')    TOPICSTR('Leeds')   DEST(QRLEEDS)
```

Figure 14. Delete and create subscriptions: fullsubs.tst

Create a cluster with two repositories. Create two partial repositories for publishing and subscribing. Rerun the script to delete everything and start again. The script also creates the topic hierarchy, and the initial wildcard subscriptions.

Note:

On other platforms, write a similar script, or type all the commands. Using a script makes it quick to delete everything and start again with an identical configuration.

```
@echo off
set port.CL1B=1421
set port.CLIA=1420
for %%A in (CL1A CL1B QMA QMB) do call :createQM %%A
call :configureQM CL1A CL1B %port.CL1B% full
call :configureQM CL1B CL1A %port.CL1A% full
for %%A in (QMA QMB) do call :configureQM %%A CL1A %port.CL1A% partial
for %%A in (topics.tst wildsubs.tst) do runmqsc QMA < %%A
for %%A in (wildsubs.tst) do runmqsc QMB < %%A
goto:eof

:createQM
echo Configure Queue manager %1
endmqm -p %1
for %%B in (dlt crt str) do %%Bmqm %1
goto:eof

:configureQM
if %1==CL1A set p=1420
if %1==CL1B set p=1421
if %1==QMA  set p=1422
if %1==QMB  set p=1423
echo configure %1 on port %p% connected to repository %2 on port %3 as %4 repository
echo DEFINE LISTENER(LST%1) TRPTYPE(TCP) PORT(%p%) CONTROL(QMGR) REPLACE | runmqsc %1
echo START LISTENER(LST%1) | runmqsc %1
if full==%4 echo ALTER  QMGR REPOS(CL1) DEADQ(SYSTEM.DEAD.LETTER.QUEUE) | runmqsc %1
echo DEFINE CHANNEL(TO.%2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('LOCALHOST(%3)') CLUSTER(CL1) REPLACE | runmqsc %1
echo DEFINE CHANNEL(TO.%1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('LOCALHOST(%p%)') CLUSTER(CL1) REPLACE | runmqsc %1
goto:eof
```

Figure 15. Create queue managers: qmgrs.bat

Update the configuration by adding the subscriptions to the cluster topics.

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

Figure 16. Update subscriptions: upsubs.bat

Run `pub.bat`, with a queue manager as a parameter, to publish messages containing the publication topic string. `Pub.bat` uses the sample program **`amqspub`**.

```
@echo off
@rem Provide queue manager name as a parameter
set S=Sports
set S=6 Sports/Football Sports/Football/Arsenal
set S=6 Sports/Rugby Sports/Rugby/Leeds
for %%B in (6) do echo %%B | amqspub %%B %1
```

Figure 17. Publish: pub.bat

Streams and topics:

Queued publish/subscribe has the concept of a publication stream that does not exist in the integrated publish/subscribe model. In queued publish/subscribe, streams provide a way of separating the flow of information for different topics. In IBM WebSphere MQ Version 6.0, a stream is implemented as a queue, defined at each broker that supports the stream. Each queue has the same name (the name of the stream). From IBM WebSphere MQ Version 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

The default stream `SYSTEM.BROKER.DEFAULT.STREAM` is set up automatically for all brokers and queue managers on a network, and no additional configuration is required to use the default stream. Think of the default stream as an unnamed default topic space. Topics published to the default stream are immediately available to all connected Version 6.0 brokers, and all queue managers from Version 7.0 onwards, with queued publish/subscribe enabled. Named streams are like separate, named, topic spaces. The named stream must be defined on each broker where it is used.

When you define a topic, the topic is available to Version 6.0 publish/subscribe brokers, and to publishers and subscribers running on a later version of IBM WebSphere MQ, with no special configuration.

If the publishers and subscribers are on different queue managers, then after the brokers are connected in the same broker hierarchy, no further configuration is required for the publications, and subscriptions to flow between them. The same interoperability works in reverse, too.

Named streams

A solution designer, working with the queued publish/subscribe programming model, might decide to place all sports publications into a named stream called `Sport`. In Version 6.0 a stream is often replicated automatically onto other brokers that use the model queue, `SYSTEM.BROKER.MODEL.STREAM`. However, for the stream to be available to a queue manager that runs on Version 7.0 onwards with queued publish/subscribe enabled, the stream must be added manually.

If you migrate a queue manager from Version 6.0, running the command **strmqbrk** migrates Version 6.0 named streams to topics. The stream `Sport` is mapped to the topic `Sport`. This is not applicable to z/OS.

Queued publish/subscribe applications that subscribe to `Soccer/Results` on stream `Sport` work without change. Integrated publish/subscribe applications that subscribe to the topic `Sport` using MQSUB, and supplying the topic string `Soccer/Results` receive the same publications too.

When the topic `Soccer/Result` is created by **strmqbrk**, it is defined as a child of the topic `Sport`, with the topic string `Sport`. A subscription to `Soccer/Results` is realized as a subscription to `Sport/Soccer/Results`, and so publications to the `Sport` stream are mapped to different place in topic space to publications to a different stream, such as `Business` .

There are scenarios for which the automatic migration performed by **strmqbrk** is not the answer, and you need to manually add streams. The task of adding a stream is described in the topic Adding a stream. You might need to add streams manually for three reasons.

1. You continue to maintain publish/subscribe applications on version 6 queue managers, which interoperate with newly written publish/subscribe applications running on later queue managers.
2. You continue to develop your queued publish/subscribe applications that are running on later version queue managers, rather than migrate the applications to the integrated publish/subscribe MQI interface.
3. The default mapping of streams to topics leads to a "collision" in topic space, and publications on a stream have the same topic string as publications from elsewhere.

Authorities

By default, at the root of the topic tree there are multiple topic objects: SYSTEM.BASE.TOPIC, SYSTEM.BROKER.DEFAULT.STREAM, and SYSTEM.BROKER.DEFAULT.SUBPOINT . Authorities (for example, for publishing or subscribing) are determined by the authorities on the SYSTEM.BASE.TOPIC; any authorities on SYSTEM.BROKER.DEFAULT.STREAM or SYSTEM.BROKER.DEFAULT.SUBPOINT are ignored. If either of SYSTEM.BROKER.DEFAULT.STREAM or SYSTEM.BROKER.DEFAULT.SUBPOINT are deleted and re-created with a non-empty topic string, authorities defined on those objects are used in the same way as a normal topic object.

Mapping between streams and topics

A queued publish/subscribe stream is mimicked in Version 7.0 onwards by creating a queue, and giving it the same name as the stream. Sometimes the queue is called the stream queue, because that is how it appears to queued publish/subscribe applications. The queue is identified to the publish/subscribe engine by adding it to the special namelist called SYSTEM.QPUBSUB.QUEUE.NAMELIST. You can add as many streams as you need, by adding additional special queues to the namelist. Finally you need to add topics, with the same names as the streams, and the same topic strings as the stream name, so you can publish and subscribe to the topics.

However, in exceptional circumstances, you can give the topics corresponding to the streams any topic strings you choose when you define the topics. The purpose of the topic string is to give the topic a unique name in the topic space. Typically the stream name serves that purpose perfectly. Sometimes, a stream name and an existing topic name collide. To resolve the problem, can choose another topic string for the topic associated with the stream. Choose any topic string, ensuring it is unique.

The topic string defined in the topic definition is prefixed in the normal way to the topic string provided by publishers and subscribers using the MQOPEN or MQSUB MQI calls. Applications referring to topics using topic objects are not affected by the choice of prefix topic string - which is why you can choose any topic string that keeps the publications unique in the topic space.

The remapping of different streams onto different topics relies on the prefixes used for the topic strings being unique, to separate one set of topics completely from another. You must define a universal topic naming convention that is rigidly adhered to for the mapping to work. In Version 7.0, if topic strings collided you might use streams to separate the topic spaces. From Version 7.0 onwards, you use the prefixing mechanism to remap a topic string to another place in topic space.

Note: When you delete a stream, delete all the subscriptions on the stream first. This action is most important if any of the subscriptions originate from other brokers in the broker hierarchy.

Example

In Figure 18 on page 37, topic 'Sport' has the topic string 'xyz' resulting in publications that originate from stream 'Sport' being prefixed with the string 'xyz' in the version 7 queue manager topic space. Publishing or subscribing in version 7 to the topic 'Sport' prefixes 'xyz' to the topic string. If the publication flows to a version 6 subscriber, the prefix 'xyz' is removed from the publication and it is placed in the 'Sport' stream. Conversely, when a publication flows from version 6 to version 7, from the 'Sport' stream to the 'Sport' topic, the prefix 'xyz' is added to the topic string.

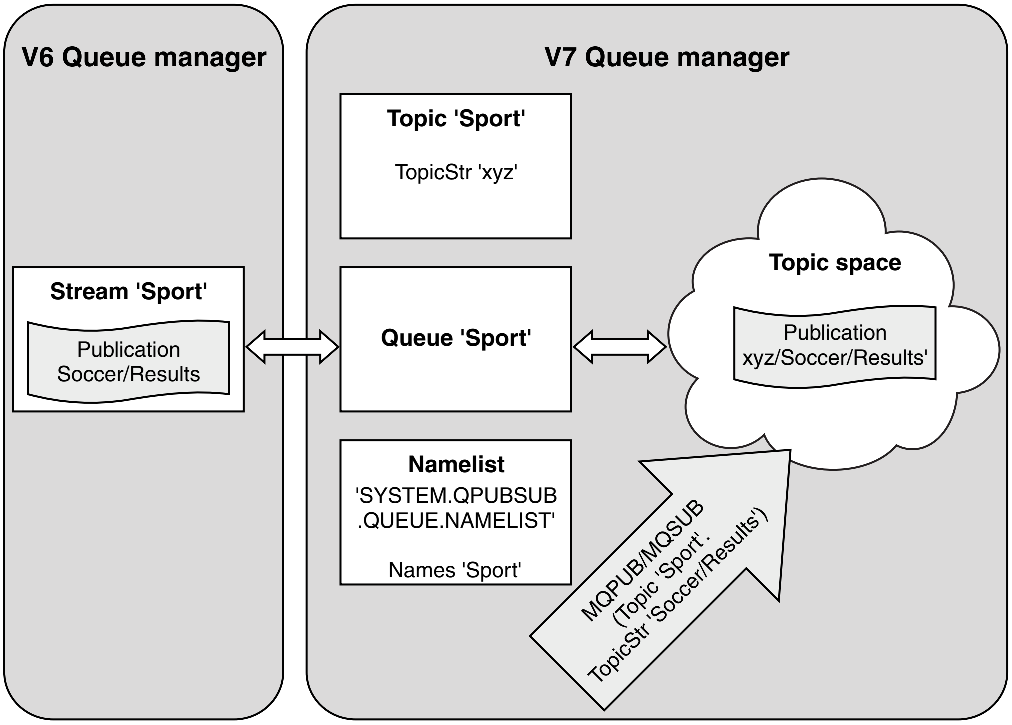


Figure 18. Version 6 streams that coexist with version 7 topics

Subscription points and topics:

A subscription point used to request publications from a particular set of publication nodes in WebSphere MQ Event Broker and Message Broker. Named subscription points are emulated by topics and topic objects.

The WebSphere MQ Event Broker V6.0 to WebSphere MQ V7.0.1 migration procedure, **migmbbrk**, converts named subscription points into topics and topic objects. A subscription point is automatically migrated if it has a retained publication, or a registered subscriber. **migmbbrk** creates topic objects from named subscription points. The name of the subscription point becomes the name of the topic object, and the topic string itself. The topic object is added to `SYSTEM.QPUBSUB.SUBPOINT.NAMELIST`.

If a topic object with the same name exists **migmbbrk** does one of two things.

1. If the topic object has a different topic string, or if the subscription point name is longer than an object name, **migmbbrk** creates a topic object with a generated name.
2. If the topic object has the same topic string, **migmbbrk** adds the existing object to the namelist.

To add subscription points manually, see Adding a subscription point.

Subscription points in WebSphere MQ Event Broker

Publication nodes are used in a WebSphere MQ Event and Message Broker message flow to filter and transmit messages to subscribers. Publishers typically do not set subscription points on publication nodes. Subscribers register an interest in a particular set of topics and typically do not specify subscription points either.

A subscription point is a way of selecting which publication nodes forward messages to a subscription. The subscriber qualifies their interest in a set of topics with the name of a subscription point.

Assign a name to the **Subscription point** property of publication node to set its subscription point name.

The subscription point property controls whether a publication to a topic is forwarded to subscribers to the same topic. Publications from publication nodes with a named subscription point are forwarded only to subscribers to the same subscription point. Publications from publication nodes without a named subscription point, the default, are forwarded only to subscribers that have not named a subscription point.

Nodes with a named subscription point send Publish command messages in MQRFH2 format, with the **SubPoint** property set. Subscriptions to a named subscription point must set **SubPoint** property in the MQRFH2 Register subscriber command message.

Subscription points in WebSphere MQ

WebSphere MQ maps subscription points to different topic spaces within the WebSphere MQ topic tree. Topics in command messages without a subscription point are mapped unchanged to the root of the WebSphere MQ topic tree and inherit properties from SYSTEM.BASE.TOPIC.

Command messages with a subscription point are processing using the list of topic objects in SYSTEM.QPUBSUB.SUBPOINT.NAMELIST. The subscription point name in the command message is matched against the topic string for each of the topic objects in the list. If a match is found, then the subscription point name is prepended, as a topic node, to the topic string. The topic inherits its properties from the associated topic object found in SYSTEM.QPUBSUB.SUBPOINT.NAMELIST.

The effect of using subscription points is to create a separate topic space for each subscription point. The topic space is rooted in a topic that has the same name as the subscription point. Topics in each topic space inherit their properties from the topic object with the same name as the subscription point.

Any properties not set in the matching topic object are inherited, in the normal fashion, from SYSTEM.BASE.TOPIC.

Existing queued publish/subscribe applications, using MQRFH2 message headers, continue to work by setting the **SubPoint** property in the Publish or Register subscriber command messages. The subscription point is combined with the topic string in the command message and the resulting topic processed like any other.

A new WebSphere MQ V7 application is unaffected by subscription points. If it uses a topic that inherits from one of the matching topic objects, it interoperates with a queued application using the matching subscription point.

Example

An existing WebSphere MQ Event Broker publish/subscribe application in a collective uses subscription points to publish share prices in different currencies. The dollar spot price of the IBM stock is published using the subscription point USD, and the topic NYSE/IBM/SPOT. The sterling price is published using the same topic, and the subscription point GBP.

The migration procedure on WebSphere MQ creates two topic objects, GBP and USD, with the corresponding topic strings 'GBP' and 'USD'.

Existing publishers to the topic NYSE/IBM/SPOT, migrated to run on WebSphere MQ, that use the subscription point USD create publications on the topic USD/NYSE/IBM/SPOT. Similarly existing subscribers to NYSE/IBM/SPOT, using the subscription point USD create subscriptions to USD/NYSE/IBM/SPOT.

Subscribe to the dollar spot price in a version 7 publish/subscribe program by calling MQSUB. Create a subscription using the USD topic object and the topic string 'NYSE/IBM/SPOT', as illustrated in the 'C' code fragment.

```
strncpy(sd.ObjectName, "USD", MQ_TOPIC_NAME_LENGTH);
sd.ObjectString.VSPtr = "NYSE/IBM/SPOT";
sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
```

Consider whether your WebSphere MQ Event Broker applications in the collective always used the subscriptions points USD and GBP. If they did, create the USD and GBP topic objects only once, as cluster topics on the cluster topic host. You do not need to carry out step 6 on page 551 of the migration procedure to change SYSTEM.BASE.TOPIC, on every queue manager in the cluster, to a cluster topic. Instead, carry out these steps:

1. Set the CLUSTER attribute of the USD and GBP topic objects on the cluster topic host.
2. Delete all the copies of the USD and GBP topic objects on other queue managers in the cluster.
3. Make sure that USD and GBP are defined in SYSTEM.QPUBSUB.SUBPOINT.NAMELIST on every queue manager in the cluster.

Distributed publish/subscribe

This section contains information about how publish/subscribe messaging can be performed between queue managers, and the two different queue manager topologies that can be used to connect queue managers, clusters and hierarchies.

Queue managers can communicate with other queue managers in your WebSphere MQ publish/subscribe system, so that subscribers can subscribe to one queue manager and receive messages that were initially published to another queue manager. This is illustrated in Figure 19 on page 40.

Figure 19 on page 40 shows a publish/subscribe system with two queue managers.

- Queue manager 2 is used by Publisher 4 to publish weather forecast information, using a topic of Weather, and information about traffic conditions on major roads, using a topic of Traffic.
- Subscriber 4 also uses this queue manager, and subscribes to information about traffic conditions using topic Traffic.
- Subscriber 3 also subscribes to information about weather conditions, even though it uses a different queue manager from the publisher. This is possible because the queue managers are linked to each other.

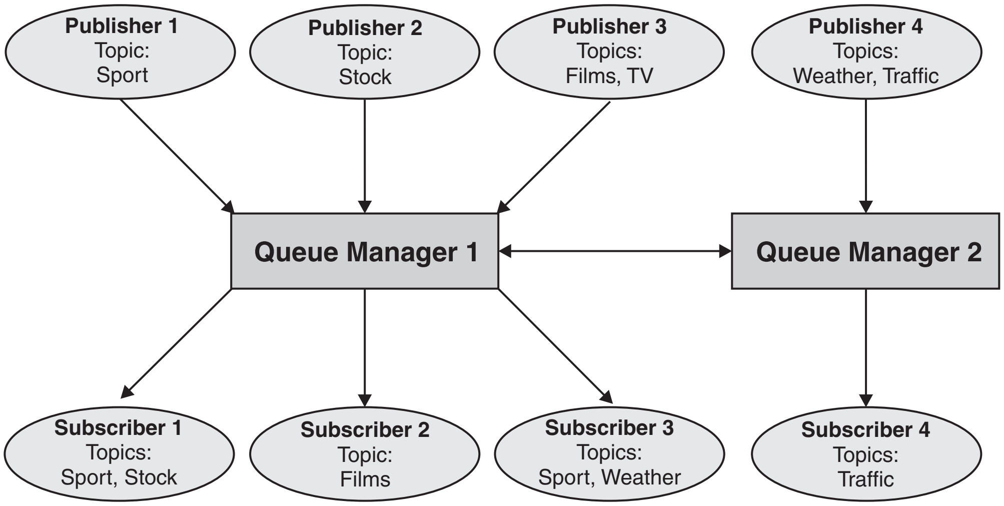


Figure 19. Publish/subscribe example with two queue managers

How does distributed publish/subscribe work?:

WebSphere MQ publish/subscribe uses proxy subscriptions to ensure that subscribers can receive messages that are published to remote queue managers.

Distributed publish/subscribe uses the same components as distributed queuing to connect networks of queue managers and consequently, the applications that connect to those queue managers. To find out more about messaging between queue managers and the components involved making connections between queue managers see the *Intercommunication* documentation.

Subscribers need not do anything beyond the standard subscription operation in a distributed publish/subscribe system. When a subscription is made on a queue manager, the queue manager manages the process by which the subscription is propagated to connected queue managers. Proxy subscriptions flow to all queue managers in the network. They are created to ensure that publications get routed back to the queue manager where the original subscription was created; see Figure 20 on page 41.

A publication is propagated to a remote queue manager only if a subscription to that topic exists on that remote queue manager.

A queue manager consolidates all the subscriptions that are created on it, whether from local applications or from remote queue managers. It creates proxy subscriptions for the topics of the subscriptions with its neighbors, unless a subscription exists; see Figure 21 on page 41.

When an application publishes information, the receiving queue manager forwards it to any applications that have valid subscriptions on remote queue managers. It might forward it through one or more intermediate queue managers; see Figure 22 on page 42.

Subscriber 1 registers a subscription for a particular topic on the Asia queue manager (1). The subscription for this topic is forwarded to all other queue managers in the network (2,3,4).

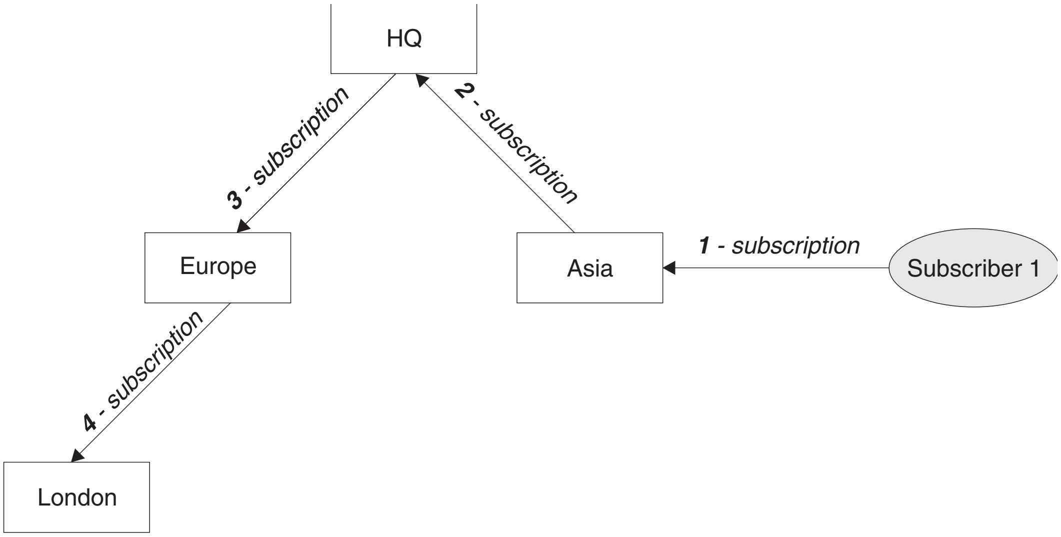


Figure 20. Propagation of subscriptions through a queue manager network

Subscriber 2 registers a subscription, to the same topic as in Figure 20, on the HQ queue manager (5). The subscription for this topic is forwarded to the Asia queue manager, so that it is aware that subscriptions exist elsewhere on the network (6). The subscription is not forwarded to the Europe queue manager, because a subscription for this topic has already been registered; see step 3 in Figure 20.

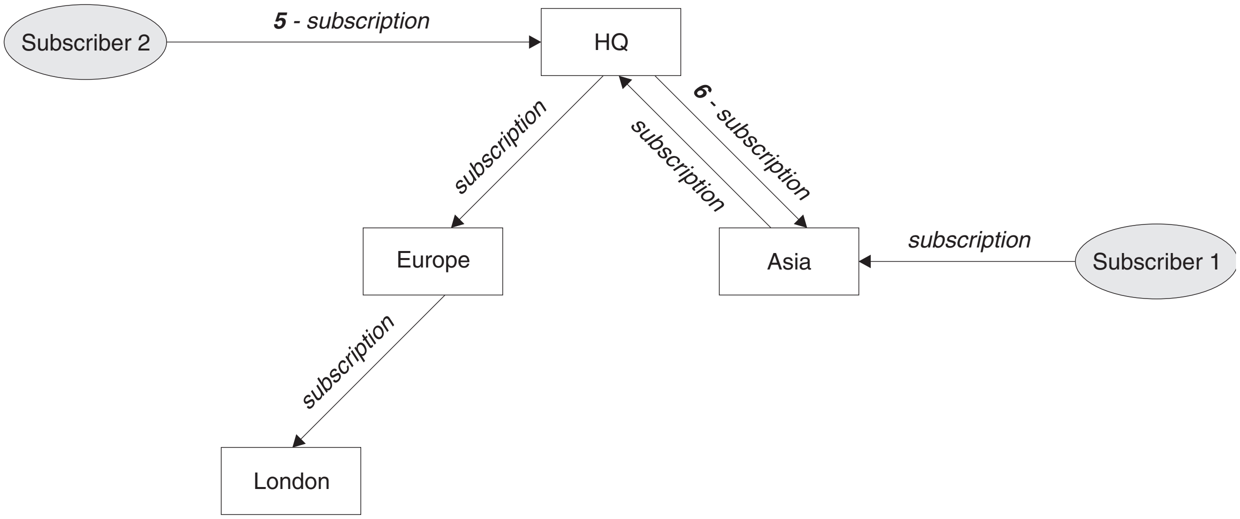


Figure 21. Multiple subscriptions

A publisher sends a publication, on the same topic as in Figure 21 on page 41, to the Europe queue manager (7). A subscription for this topic exists from HQ to Europe, so the publication is forwarded to the HQ queue manager (8). However, no subscription exists from London to Europe (only from Europe to London), so the publication is not forwarded to the London queue manager. The HQ queue manager sends the publication directly to subscriber 2 and to the Asia queue manager (9). The publication is forwarded to subscriber 1 from Asia (10).

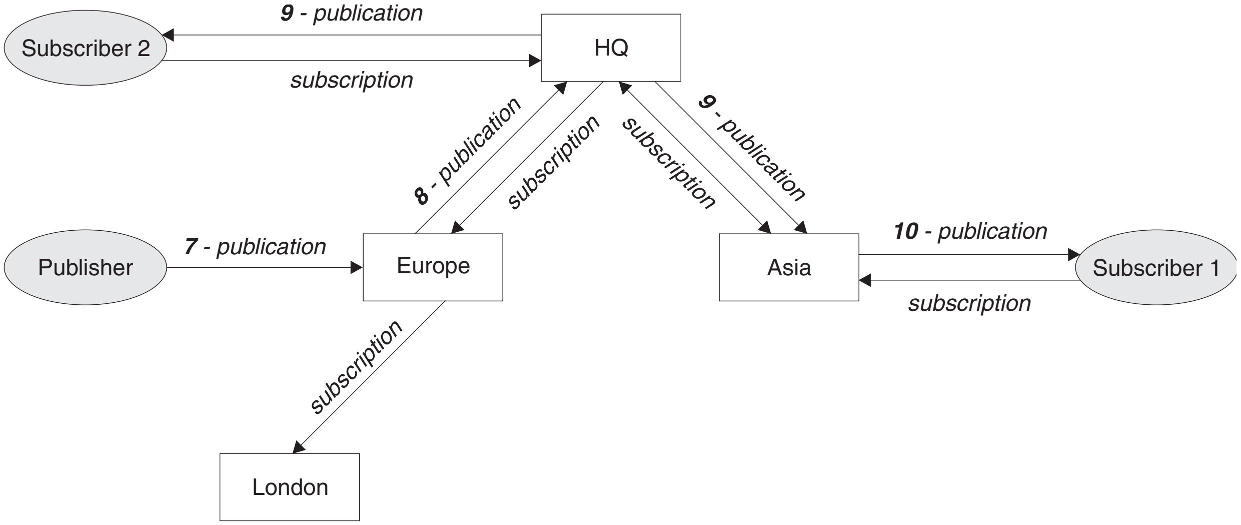


Figure 22. Propagation of publications through a queue manager network

When a queue manager sends any publications or subscriptions to another queue manager, it sets its own user ID in the message. If you are using a publish/subscribe hierarchy, and if the incoming channel is set up to put messages with the authority of the user ID in the message, then you must authorize the user ID of the sending queue manager; see “Using default user IDs with a queue manager hierarchy” on page 99. If you are using publish/subscribe clusters, authorization is handled by the cluster.

Because of the interconnected nature of publish/subscribe queue managers, it takes time for proxy subscriptions to propagate around all nodes in the network. Remote publications do not necessarily start being subscribed to immediately. You can eliminate the subscription delay by using the **Topic** attribute PROXYSUB with the value FORCE as described in “More on routing mechanisms” on page 44.

The subscription operation finishes when the proxy subscription has been put on the appropriate transmission queue for each directly connected queue manager. The subscription operation does not wait for the propagation of the proxy subscription to the rest of the topology.

Proxy subscriptions are associated with the queue manager name that created them. If queue managers in the hierarchy have the same name, it can result in publications failing to reach them. To avoid this problem, as with point-to-point messaging, give queue managers unique names, especially if they are directly or indirectly connected in a WebSphere MQ network.

Within a distributed publish/subscribe network the flow of publications and subscriptions can be controlled, and if appropriate, restricted, using publication and subscription scope.

Proxy subscription aggregation and publication aggregation:

Distributed publish/subscribe publications and proxy subscriptions are aggregated to minimize the quantity of messages passing between publish/subscribe queue managers.

A proxy subscription is a subscription made by one queue manager for topics published on another queue manager. You do not create proxy subscriptions explicitly, the queue manager does so on your behalf; see “How does distributed publish/subscribe work?” on page 40.

You can connect queue managers together into a publish/subscribe hierarchy, or into a publish/subscribe cluster. Proxy subscriptions flow between the connected queue managers. Proxy subscriptions cause publications to a topic created by a publisher connected to one queue manager to be received by subscribers to that topic connected to other queue managers; see “Publish/subscribe topologies” on page 53.

A proxy subscription flows between queue managers for each individual topic string that is subscribed to by a subscription.

You can restrict the flow of proxy subscriptions and publications between connected queue managers using the **Topic** attributes PUBSCOPE and SUBSCOPE. You can also restrict the flow of proxy subscriptions containing wildcards by setting the **Topic** attribute WILDCARD to BLOCK; see “Wildcard rules” on page 45.

Proxy subscriptions are flowed between queue managers asynchronously to the creation of subscriptions. You can reduce the latency of waiting for a proxy subscription to be propagated to all the connected queue managers, by setting the **Topic** attribute PROXYSUB to FORCE on the topic, or a parent of a topic that will be subscribed to; see “More on routing mechanisms” on page 44.

Proxy subscription aggregation

Proxy subscriptions are aggregated using a duplicate elimination system. For a particular resolved topic string, a proxy subscription is sent on the first local subscription or received proxy subscription. Subsequent subscriptions to the same topic string make use of this existing proxy subscription.

The proxy subscription is canceled after the last local subscription or received proxy subscription is canceled.

In publish/subscribe topologies with many thousands of subscriptions to individual topic strings, or where the existence of those subscriptions may be rapidly changing, the overhead of the proxy subscription propagation must be considered. The individual proxy subscriptions can be consolidated through the use of the topic attribute **PROXYSUB** being set to FORCE. For more details about routing mechanisms and cluster topic performance, see “More on routing mechanisms” on page 44.

Publication aggregation

When there is more than one subscription to the same topic string on a queue manager, only a single copy of each publication matching that topic string is sent from other queue managers in the publish/subscribe topology. On arrival of the message, the local queue manager delivers a copy of the message to each matching subscription.

It is possible for more than one proxy subscription to match the topic string of a single publication when the proxy subscriptions contain wildcards. If a message is published on a queue manager that matches two or more proxy subscriptions created by a single connected queue manager, only one copy of the publication is forwarded to the remote queue manager to satisfy the multiple proxy subscriptions.

More on routing mechanisms:

Publish everywhere is an alternative routing mechanism to individual proxy subscription-forwarding. Individual proxy subscription forwarding means that only publications that have a matching subscription on the topic string are sent to a remote messaging server. Publish everywhere, or broadcast, works by forwarding all publications that are published to a messaging server, to all other messaging servers in a distributed publish/subscribe network. The receiving messaging servers then deliver those publications that match local subscriptions.

Each mechanism has its merits, but also has limitations.

Individual proxy subscription forwarding

This mechanism results in the least amount of inter-queue manager publication traffic as only those publications that match subscriptions on a queue manager are sent.

However:

- Each individual topic string that is subscribed to results in a proxy subscription that is sent to all other queue managers in the publish/subscribe topology. This messaging overhead can be significant if there are many thousands of subscriptions to create or delete (for example, all non-durable subscriptions after a restart of a queue manager) or if the set of subscriptions is rapidly changing, and each is to a different topic string.
- Proxy subscriptions are flowed to other queue managers using asynchronous messaging, therefore, there is a delay between the creation of a subscription and proxy subscription creation, delivery, and processing by the other queue managers. Messages that are published at those queue managers in that interval are not delivered to the remote subscription.

Publish everywhere

With this mechanism:

- There is no per topic string proxy subscription overhead on the system which means rapid subscription creation, deletion, or change does not result in increased network load and processing.
- There is no delay between creating a subscription and publications being flowed to a queue manager as they are always flowed to all queue managers. Therefore, there is no window where publications are not delivered to newly created remote subscriptions.

However:

- All publications are sent to all queue managers in the publish/subscribe topology, potentially resulting in excessive network traffic where publications do not have matching subscriptions on each queue manager.

You might want to use the publish everywhere mechanism when you expect a publication to be subscribed to from a significant proportion of your queue managers in the cluster or hierarchy, or where the proxy subscription overheads are too great because of the frequency of subscription changes. This method of working might be more effective in these instances than in others where you experience increased messaging traffic when publications are sent to all queue managers, rather than to the queue managers with matching subscriptions.

A publish everywhere mechanism can be enabled in IBM WebSphere MQ distributed publish/subscribe topologies by setting the **PROXYSUB** attribute to FORCE for a high-level topic object.

For details about disabling individual proxy subscriptions, see “Disabling individual proxy subscriptions” on page 67.

When this forced proxy subscription is propagated throughout the topology, any new subscriptions immediately receive any publications from other connected queue managers, without suffering latency.

Care must be taken when you are configuring such a system. No topic objects below a topic with **PROXYSUB** set to FORCE must be in a different cluster or hierarchy stream to the node where **PROXYSUB** is set to FORCE. Similarly, lower topic objects must not set their **WILDCARD** attribute to BLOCK. In both cases this can result in published messages not flowing from one queue manager to another correctly.

Even when **PROXYSUB** is set to FORCE, a proxy subscription for each individual topic string that is subscribed to continues to be propagated. If the number and frequency of subscriptions is high enough to cause significant overhead to the system, they can be disabled for all topics on a queue manager. For details about disabling individual proxy subscriptions, see "Disabling individual proxy subscriptions" on page 67.

Multicast and subscription latency

Subscription latency and the PROXYSUB(FORCE) option can be used to maintain a proxy subscription.

For example, there is the potential problem of a proxy subscription from QM_B to QM_A being unmade after all the subscribers are disconnected. This situation might not be wanted if you require the multicast traffic to continue even when the unicast connection to the queue manager terminates. Multicast for WebSphere MQ maintains the proxy subscription for a short time, in case a new subscriber connects, by adding a few minutes latency to each proxy subscription so that they are not unmade the instant that the last subscriber terminates.

You can also use the PROXYSUB(FORCE) option on the topic to ensure that an outstanding proxy subscription is always outstanding. You must ensure that the messages flowing across the queues are required by at least one subscriber for most of the time that the subscription is active. If PROXYSUB(FORCE) is set, a proxy subscription might be sent before the first local subscription or received proxy subscription, and will not be canceled even after the last local subscription or received proxy subscription is canceled.

If the subscription is still unmade, peer-to-peer communication can be used to ensure that message transfers continue; For more information, see High availability for multicast .

Wildcard rules:

Wildcards in proxy subscriptions are converted to use topic wildcards.

If a subscription for a wildcard is received, it can be a character, as used by WebSphere MQ Version 6.0. It can also be a topic, as used by WebSphere Message Broker Version 6.0 and WebSphere MQ Version 7.0.

- Character wildcards use `*` to represent any character, including `/`.
- Topic wildcards use `#` to represent a portion of the topic space between `/` characters.

In WebSphere MQ Version 7.0, all proxy subscriptions are converted to use topic wildcards. If a character wildcard is found, it is replaced with a `#` character, back to the nearest `/`. For example, `/aaa/bbb/c*d` is converted to `/aaa/bbb/#`. The conversion results in remote queue managers sending slightly more publications than were explicitly subscribed to. The additional publications are filtered out by the local queue manager, when it delivers the publications to its local subscribers.

Controlling wildcard usage with the WILDCARD property

Use the MQSC **`Topic`** WILDCARD property or the equivalent PCF Topic `WildcardOperation` property to control the delivery of publications to subscriber applications that use wildcard topic string names. The WILDCARD property can have one of two possible values:

WILDCARD
The behavior of wildcard subscriptions with respect to this topic.

PASSTHRU

Subscriptions made to a wildcarded topic less specific than the topic string at this topic object receive publications made to this topic and to topic strings more specific than this topic.

BLOCK

Subscriptions made to a wildcarded topic less specific than the topic string at this topic object do not receive publications made to this topic or to topic strings more specific than this topic.

The value of this attribute is used when subscriptions are defined. If you alter this attribute, the set of topics covered by existing subscriptions is not affected by the modification. This scenario applies also if the topology is changed when topic objects are created or deleted; the set of topics matching subscriptions created following the modification of the WILDCARD attribute is created using the modified topology. If you want to force the matching set of topics to be re-evaluated for existing subscriptions, you must restart the queue manager.

In the example, "Example: Create the Sport publish/subscribe cluster" on page 32, you can follow the steps to create the topic tree structure shown in Figure 10 on page 28.

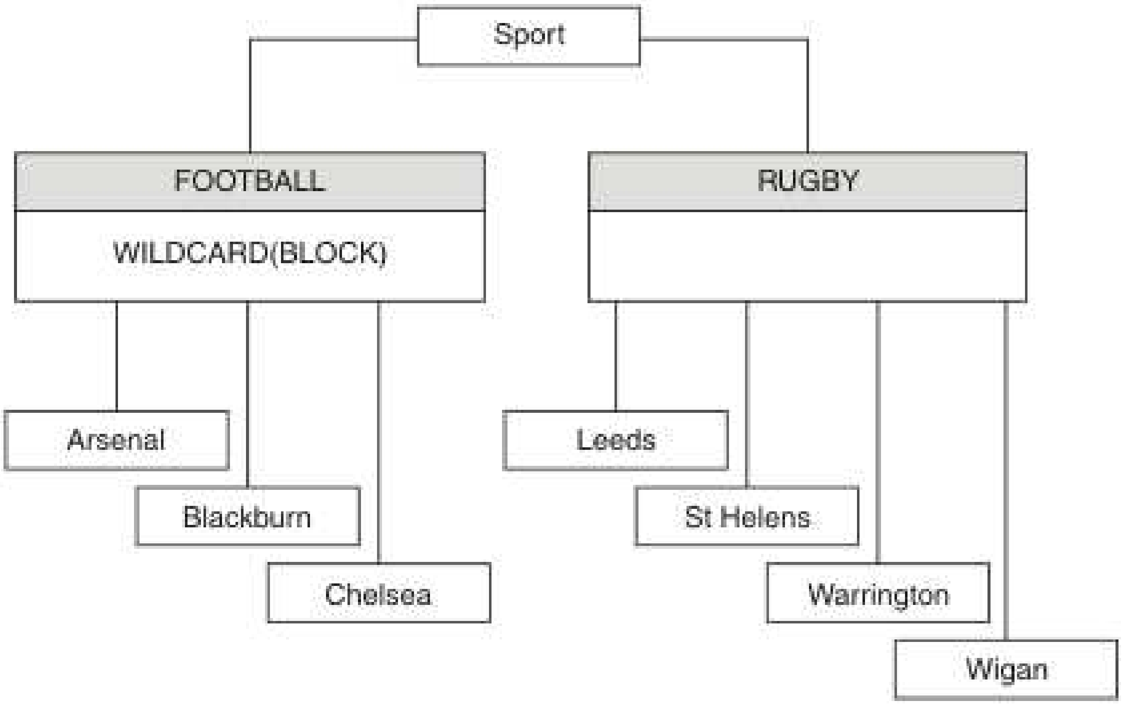


Figure 23. A topic tree that uses the WILDCARD property, `BLOCK`

A subscriber using the wildcard topic string # receives all publications to the `Sport` topic and the `Sport/Rugby` subtree. The subscriber receives no publications to the `Sport/Football` subtree, because the `WILDCARD` property value of the `Sport/Football` topic is `BLOCK`.

`PASSTHRU` is the default setting. You can set the `WILDCARD` property value `PASSTHRU` to nodes in the `Sport` tree. If the nodes do not have the `WILDCARD` property value `BLOCK`, setting `PASSTHRU` does not alter the behavior observed by subscribers to nodes in the `Sports` tree.

In the example, create subscriptions to see how the wildcard setting affects the publications that are delivered; see Figure 14 on page 33. Run the publish command in Figure 17 on page 34 to create some publications.

```
pub QMA
```

Figure 24. Publish to QMA

The results are shown in Table 3 on page 29. Notice how setting the WILDCARD property value BLOCK, prevents subscriptions with wildcards from receiving publications to topics within the scope of the wildcard.

Table 6. Publications received on QMA

Subscription	Topic string	Publications received	Notes
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football prevents wildcard subscription on Arsenal
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Default WILDCARD on Sports/Rugby does not prevent wildcard subscription on Leeds.

Note:

Suppose a subscription has a wildcard that matches a topic object with the WILDCARD property value BLOCK. If the subscription also has a topic string to the right of the matching wildcard, the subscription never receives a publication. The set of publications that are not blocked are publications to topics that are parents of the blocked wildcard. Publications to topics that are children of the topic with the BLOCK property value are blocked by the wildcard. Therefore subscription topic strings that include a topic to the right of the wildcard never receive any publications to match.

Setting the WILDCARD property value to BLOCK does not mean you cannot subscribe using a topic string that includes wildcards. Such a subscription is normal. The subscription has an explicit topic that matches the topic with a topic object having a WILDCARD property value BLOCK. It uses wildcards for topics that are parents or children of the topic with the WILDCARD property value BLOCK. In the example in Figure 10 on page 28, a subscription such as Sports/Football/# can receive publications.

Wildcards and cluster topics

Cluster topic definitions are propagated to every queue manager in a cluster. A subscription to a cluster topic at one queue manager in a cluster results in the queue manager creating proxy subscriptions. A proxy subscription is created at every other queue manager in the cluster. Subscriptions using topics strings containing wildcards, combined with cluster topics, can give hard to predict behavior. The behavior is explained in the following example.

In the cluster set up for the example, “Example: Create the Sport publish/subscribe cluster” on page 32, QMB has the same set of subscriptions as QMA, yet QMB received no publications after the publisher published to QMA, see Figure 11 on page 28. Although the Sports/Football and Sports/Rugby topics are cluster topics, the subscriptions defined in fullsubs.tst do not reference a cluster topic. No proxy subscriptions are propagated from QMB to QMA. Without proxy subscriptions, no publications to QMA are forwarded to QMB .

Some of the subscriptions, such as Sports/#/Leeds, might seem to reference a cluster topic, in this case Sports/Rugby . The Sports/#/Leeds subscription actually resolves to the topic object SYSTEM.BASE.TOPIC.

The rule for resolving the topic object referenced by a subscription such as, Sports/#/Leeds is as follows. Truncate the topic string to the first wildcard. Scan left through the topic string looking for the first topic that has an associated administrative topic object. The topic object might specify a cluster name, or define a local topic object. In the example, Sports/#/Leeds, the topic string after truncation is Sports, which has no topic object, and so Sports/#/Leeds inherits from SYSTEM.BASE.TOPIC, which is a local topic object.

To see how subscribing to clustered topics can change the way wildcard propagation works, run the batch script, upsubs.bat. The script clears the subscription queues, and adds the cluster topic subscriptions in fullsubs.tst. Run puba.bat again to create a batch of publications; see Figure 11 on page 28.

Table 4 on page 30 shows the result of adding two new subscriptions to the same queue manager that the publications were published on. The result is as expected, the new subscriptions receive one publication each, and the numbers of publications received by the other subscriptions are unchanged. The unexpected results occur on the other cluster queue manager; see Table 5 on page 30.

Table 7. Publications received on QMA

Subscription	Topic string	Publications received	Notes
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football prevents wildcard subscription on Arsenal
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Default WILDCARD on Sports/Rugby does not prevent wildcard subscription on Leeds.
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal receives a publication because the subscription does not have a wildcard.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds would receive a publication in any event.

Table 5 on page 30 shows the results of adding the two new subscriptions on QMB and publishing on QMA. Recall that QMB received no publications without these two new subscriptions. As expected, the two new subscriptions receive publications, because Sports/FootBall and Sports/Rugby are both cluster topics. QMB forwarded proxy subscriptions for Sports/Football/Arsenal and Sports/Rugby/Leeds to QMA, which then sent the publications to QMB.

The unexpected result is that the two subscriptions Sports/# and Sports/#/Leeds that previously received no publications, now receive publications. The reason is that the Sports/Football/Arsenal and Sports/Rugby/Leeds publications forwarded to QMB for the other subscriptions are now available for any subscriber attached to QMB . Consequently the subscriptions to the local topics Sports/# and Sports/#/Leeds receive the Sports/Rugby/Leeds publication. Sports/#/Arsenal continues not to receive a publication, because Sports/Football has its WILDCARD property value set to BLOCK.

Table 8. Publications received on QMB

Subscription	Topic string	Publications received	Notes
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football*
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football prevents wildcard subscription on Arsenal
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Default WILDCARD on Sports/Rugby does not prevent wildcard subscription on Leeds .*
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal receives a publication because the subscription does not have a wildcard.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds would receive a publication in any event.

In most applications, it is undesirable for one subscription to influence the behavior of another subscription. One important use of the WILDCARD property with the value BLOCK is to make the subscriptions to the same topic string containing wildcards behave uniformly. Whether the subscription is on the same queue manager as the publisher, or a different one, the results of the subscription are the same.

Wildcards and streams

WebSphere MQ Version 6 streams are mapped to topics by WebSphere MQ Version 7; see “Streams and topics” on page 35 . In the default mapping, which is performed by **strmqbrk** in Version 7, all the topics in the stream Sports would be mapped to the topic Sports. All the topics in the stream Business would be mapped to the topic Business.

A subscription in WebSphere MQ Version 6 to * in the Sports stream receives all publications in the Sports tree, and no publications in the Business tree. The same subscription in version 7 would receive all the publications in the Sports tree and all the publications in the Business tree. To block this behavior, when streams are migrated to version 7, **strmqbrk** sets the WILDCARD property. It sets it to the value BLOCK for each of the top-level topics that are migrated from a stream. The WILDCARD property of Sports and Business is set to the value BLOCK by the conversion from the version 6 streams called Sports and Business.

For a new application written to the publish/subscribe API, the effect is that a subscription to * receives no publications. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

The behavior of an existing queued publish/subscribe application does not change when the publish/subscribe broker is migrated to WebSphere MQ Version 7. The **StreamName** property in the **Publish**, **Register Publisher**, or **Subscriber** commands is mapped to the name of the topic the stream has been migrated to.

Wildcards and subscription points

WebSphere Message Broker subscriptions points are mapped to topics by WebSphere MQ Version 7; see “Subscription points and topics” on page 37. In the default mapping, which is performed by **migmqbrk** in Version 7, all the topics in the subscription point Sports would be mapped to the topic Sports. All the topics in the subscription point Business would be mapped to the topic Business.

A subscription on WebSphere Message Broker Version 6 to * in the Sports subscription point receives all publications in the Sports tree, and no publications in the Business tree. The same subscription in version

7 would receive all the publications in the Sports tree and all the publications in the Business tree. To block this behavior, when subscription points are migrated to version 7, **migmqbrk** sets the WILDCARD property. It sets it to the value BLOCK for each of the top-level topics that are migrated from a subscription point. The WILDCARD property of Sports and Business is set to the value BLOCK by the conversion from the WebSphere Message Broker subscription points called Sports and Business .

For a new application written to the publish/subscribe API, the effect of the migration is that a subscription to * receives no publications. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

The behavior of an existing queued publish/subscribe application does not change when the publish/subscribe broker is migrated to WebSphere MQ Version 7. The **SubPoint** property in the **Publish**, **Register Publisher**, or **Subscriber** commands is mapped to the name of the topic the subscription has been migrated to.

Example: Create the Sport publish/subscribe cluster

The steps that follow create a cluster, CL1, with four queue managers: two full repositories, CL1A and CL1B, and two partial repositories, QMA and QMB. The full repositories are used to hold only cluster definitions. QMA is designated the cluster topic host. Durable subscriptions are defined on both QMA and QMB.

Note: The example is coded for Windows. You must recode Create qmgrs.bat and create pub.bat to configure and test the example on other platforms.

1. Create the script files.
   a. Create topics.tst
   b. Create wildsubs.tst
   c. Create fullsubs.tst
   d. Create qmgrs.bat
   e. create pub.bat
2. Run Create qmgrs.bat to create the configuration.

   ```
   qmgrs
   ```

Create the topics in Figure 10 on page 28. The script in figure 5 creates the cluster topics Sports/Football and Sports/Rugby.

Note: The REPLACE option does not replace the TOPICSTR properties of a topic. TOPICSTR is a property that is usefully varied in the example to test different topic trees. To change topics, delete the topic first.

```
DELETE TOPIC ('Sports')
DELETE TOPIC ('Football')
DELETE TOPIC ('Arsenal')
DELETE TOPIC ('Blackburn')
DELETE TOPIC ('Chelsea')
DELETE TOPIC ('Rugby')
DELETE TOPIC ('Leeds')
DELETE TOPIC ('Wigan')
DELETE TOPIC ('Warrington')
DELETE TOPIC ('St. Helens')

DEFINE TOPIC ('Sports')     TOPICSTR('Sports')
DEFINE TOPIC ('Football')   TOPICSTR('Sports/Football') CLUSTER(CL1) WILDCARD(BLOCK)
DEFINE TOPIC ('Arsenal')    TOPICSTR('Sports/Football/Arsenal')
DEFINE TOPIC ('Blackburn')  TOPICSTR('Sports/Football/Blackburn')
DEFINE TOPIC ('Chelsea')    TOPICSTR('Sports/Football/Chelsea')
DEFINE TOPIC ('Rugby')      TOPICSTR('Sports/Rugby')    CLUSTER(CL1)
DEFINE TOPIC ('Leeds')      TOPICSTR('Sports/Rugby/Leeds')
DEFINE TOPIC ('Wigan')      TOPICSTR('Sports/Rugby/Wigan')
DEFINE TOPIC ('Warrington') TOPICSTR('Sports/Rugby/Warrington')
DEFINE TOPIC ('St. Helens') TOPICSTR('Sports/Rugby/St. Helens')
```

Figure 25. Delete and create topics: topics.tst

Note: Delete the topics, as REPLACE does not replace topic strings.

Create subscriptions with wildcards. The wildcards corresponding the topics with topic objects in Figure 10 on page 28. Create a queue for each subscription. The queues are cleared and the subscriptions deleted when the script is run or rerun.

Note: The REPLACE option does not replace TOPICOBJ or TOPICSTR properties of a subscription. TOPICOBJ or TOPICSTR are the properties that are usefully varied in the example to test different subscriptions. To change them, delete the subscription first.

```
DEFINE QLOCAL(QSPORTS)   REPLACE
DEFINE QLOCAL(QSARSENAL) REPLACE
DEFINE QLOCAL(QSLEEDS)   REPLACE
CLEAR  QLOCAL(QSPORTS)
CLEAR  QLOCAL(QSARSENAL)
CLEAR  QLOCAL(QSLEEDS)

DELETE SUB   (SPORTS)
DELETE SUB   (SARSENAL)
DELETE SUB   (SLEEDS)
DEFINE SUB   (SPORTS)   TOPICSTR('Sports/#')          DEST(QSPORTS)
DEFINE SUB   (SARSENAL) TOPICSTR('Sports/+/Arsenal')  DEST(QSARSENAL)
DEFINE SUB   (SLEEDS)   TOPICSTR('Sports/+/Leeds')    DEST(QSLEEDS)
```

Figure 26. Create wildcard subscriptions: wildsubs.tst

Create subscriptions that reference the cluster topic objects.

Note:

The delimiter, /, is automatically inserted between the topic string referenced by TOPICOBJ, and the topic string defined by TOPICSTR.

The definition, DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) creates the same subscription. TOPICOBJ is used as a quick way to reference topic string you have already defined. The subscription, when created, no longer refers to the topic object.

```
DEFINE QLOCAL(QFARSENAL) REPLACE
DEFINE QLOCAL(QRLEEDS)   REPLACE
CLEAR  QLOCAL(QFARSENAL)
CLEAR  QLOCAL(QRLEEDS)

DELETE SUB   (FARSENAL)
DELETE SUB   (RLEEDS)
DEFINE SUB   (FARSENAL) TOPICOBJ('Football') TOPICSTR('Arsenal') DEST(QFARSENAL)
DEFINE SUB   (RLEEDS)   TOPICOBJ('Rugby')    TOPICSTR('Leeds')   DEST(QRLEEDS)
```

Figure 27. Delete and create subscriptions: fullsubs.tst

Create a cluster with two repositories. Create two partial repositories for publishing and subscribing. Rerun the script to delete everything and start again. The script also creates the topic hierarchy, and the initial wildcard subscriptions.

Note:

On other platforms, write a similar script, or type all the commands. Using a script makes it quick to delete everything and start again with an identical configuration.

```
@echo off
set port.CL1B=1421
set port.CLIA=1420
for %%A in (CL1A CL1B QMA QMB) do call :createQM %%A
call :configureQM CL1A CL1B %port.CL1B% full
call :configureQM CL1B CL1A %port.CL1A% full
for %%A in (QMA QMB) do call :configureQM %%A CL1A %port.CL1A% partial
for %%A in (topics.tst wildsubs.tst) do runmqsc QMA < %%A
for %%A in (wildsubs.tst) do runmqsc QMB < %%A
goto:eof

:createQM
echo Configure Queue manager %1
endmqm -p %1
for %%B in (dlt crt str) do %%Bmqm %1
goto:eof

:configureQM
if %1==CL1A set p=1420
if %1==CL1B set p=1421
if %1==QMA  set p=1422
if %1==QMB  set p=1423
echo configure %1 on port %p% connected to repository %2 on port %3 as %4 repository
echo DEFINE LISTENER(LST%1) TRPTYPE(TCP) PORT(%p%) CONTROL(QMGR) REPLACE | runmqsc %1
echo START LISTENER(LST%1) | runmqsc %1
if full==%4 echo ALTER  QMGR REPOS(CL1) DEADQ(SYSTEM.DEAD.LETTER.QUEUE) | runmqsc %1
echo DEFINE CHANNEL(TO.%2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('LOCALHOST(%3)') CLUSTER(CL1) REPLACE | runmqsc %1
echo DEFINE CHANNEL(TO.%1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('LOCALHOST(%p%)') CLUSTER(CL1) REPLACE | runmqsc %1
goto:eof
```

Figure 28. Create queue managers: qmgrs.bat

Update the configuration by adding the subscriptions to the cluster topics.

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

Figure 29. Update subscriptions: upsubs.bat

Run `pub.bat`, with a queue manager as a parameter, to publish messages containing the publication topic string. `Pub.bat` uses the sample program **`amqspub`**.

```
@echo off
@rem Provide queue manager name as a parameter
set S=Sports
set S=6 Sports/Football Sports/Football/Arsenal
set S=6 Sports/Rugby Sports/Rugby/Leeds
for %%B in (6) do echo %%B | amqspub %%B %1
```

Figure 30. Publish: pub.bat

Publish/subscribe topologies:

A *publish/subscribe topology* consists of queue managers and the connections between them, that support publish/subscribe applications.

A publish/subscribe application can consist of a network of queue managers connected together. The queue managers can all be on the same physical system, or they can be distributed over several physical systems. By connecting queue managers together, publications can be received by an application using any queue manager in the network.

This provides the following benefits:
- Client applications can communicate with a nearby queue manager rather than with a distant queue manager, thereby getting better response times.
- By using more than one queue manager, more subscribers can be supported.

You can arrange queue managers that are doing publish/subscribe messaging in two different ways, clusters and hierarchies. For examples of a simple cluster and a simple hierarchy, see Figure 31 and Figure 32 on page 54. For more information about these two topologies and to find out which is most appropriate for you, refer to the information in this section of the product documentation.

It is possible to use both topologies in combination by joining clusters together in a hierarchy.

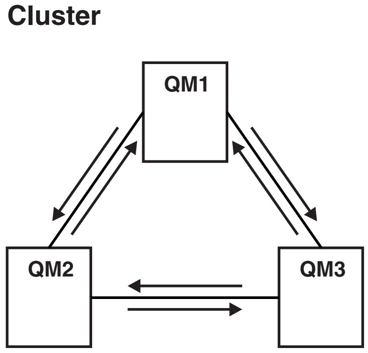


Figure 31. Simple publish/subscribe cluster

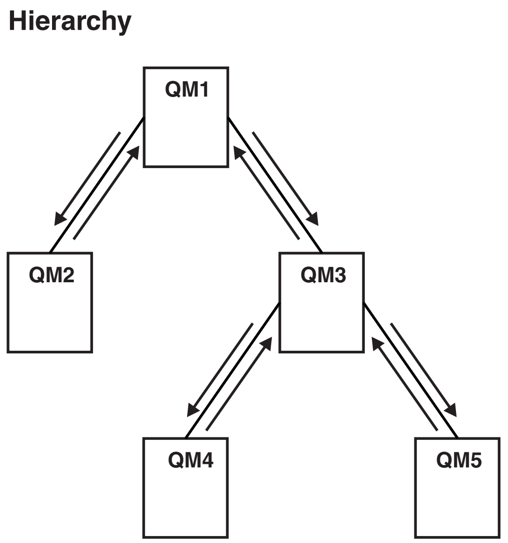


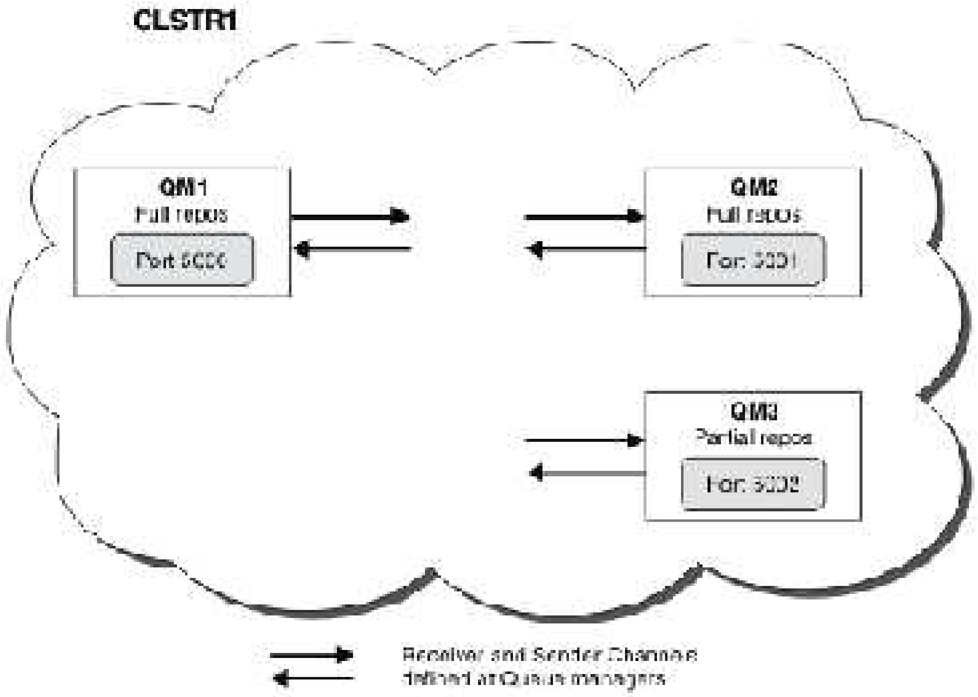
Figure 32. Simple publish/subscribe hierarchy

Setting up a publish/subscribe cluster: Scenario 1:

Add two queue managers as full repositories to a cluster, and define the channels between them.

About this task

The following diagram has three queue managers; QM1, QM2, and QM3:



QM1 and QM2 are full repositories in a cluster, and QM3 is a partial repository.

Scenario 1 adds QM1 and QM2 to the cluster DEMO as full repositories.

Scenario 2 adds QM3 to the cluster DEMO as a partial repository.

These tasks require at least one command window.

Procedure

1. Set QM1 and QM2 as full repositories of the DEMO cluster:

   ```
   alter QMGR REPOS(DEMO)
   ```

2. Define and start a listener for QM1:

   ```
   define listener(QM1_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5000)
   start listener(QM1_LS)
   ```

3. Define and start a listener for QM2:

   ```
   define listener(QM2_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5001)
   start listener(QM2_LS)
   ```

4. Define a receiver channel for QM1:

   ```
   DEFINE CHANNEL(DEMO.QM1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)') CLUSTER(DEMO) DESCR('TCP Cluster-
   ```

5. Define a sender channel from QM1 to QM2:

   ```
   DEFINE CHANNEL(DEMO.QM2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5001)') CLUSTER(DEMO) DESCR('TCP Cluster-s
   ```

6. Define a receiver channel for QM2:

   ```
   DEFINE CHANNEL(DEMO.QM2) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5001)') CLUSTER(DEMO) DESCR('TCP Cluster-
   ```

7. Define a sender channel from QM2 to QM1:

   ```
   DEFINE CHANNEL(DEMO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)') CLUSTER(DEMO) DESCR('TCP Cluster-s
   ```

8. Define the cluster topic scores on QM1:

   ```
   define topic(scores) TOPICSTR(/football) CLUSTER(DEMO)
   ```

9. Verify the setup with the following commands:

   ```
   display topic(scores) type(all) clusinfo
   display clusqmgr(*)
   display chstatus(*)
   ```

10. Test the setup using two command windows:

    a. Enter this command in the first command window:

       ```
       /opt/mqm/samp/bin/amqspub /FOOTBALL/scores QM1
       ```

    b. Enter this command in the second command window:

       ```
       /opt/mqm/samp/bin/amqssub /FOOTBALL/scores QM2
       ```

Related information:

Managing WebSphere MQ clusters
You can create, extend, and maintain IBM WebSphere MQ clusters.

Setting up a new cluster
Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Setting up a publish/subscribe cluster: Scenario 2:

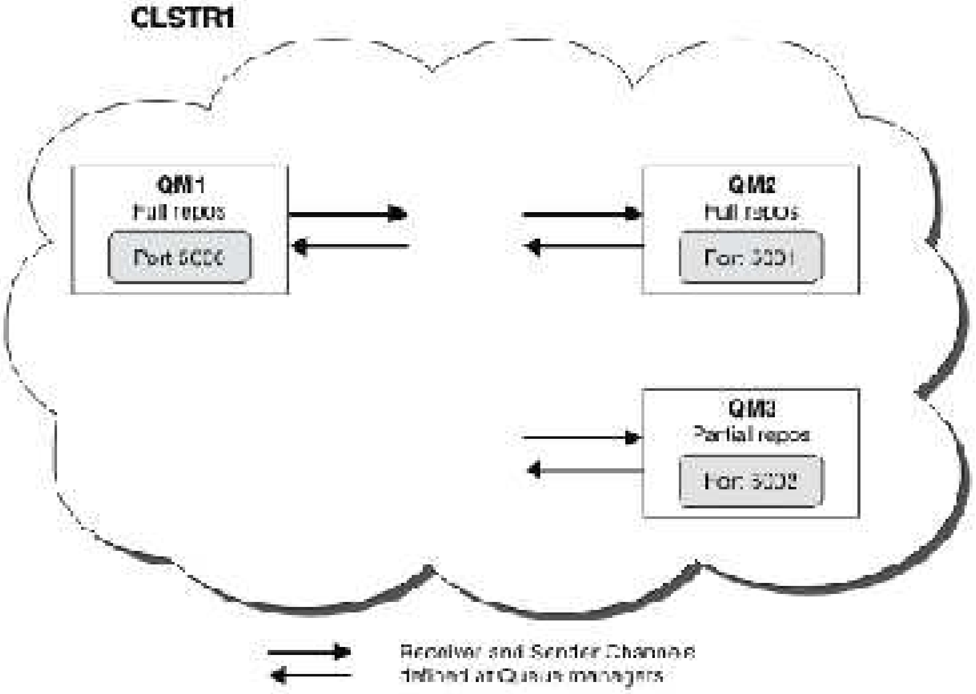
Add a third queue manager as a partial repository to the cluster.

Before you begin

You must have completed the task in "Setting up a publish/subscribe cluster: Scenario 1" on page 54 before completing this task.

About this task

The following diagram has 3 queue managers; QM1, QM2, and QM3:



QM1 and QM2 are full repositories in a cluster, and QM3 is a partial repository.

Scenario 1 adds QM1 and QM2 to the cluster DEMO as full repositories.

Scenario 2 adds QM3 to the cluster DEMO as a partial repository.

These tasks require at least 1 command window.

Procedure

1. Define and start a listener for QM3:
   ```
   define listener(QM3_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5002)
   start listener(QM3_LS)
   ```
2. Define a receiver channel for QM3:
   ```
   DEFINE CHANNEL(DEMO.QM3) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5002)') CLUSTER (DEMO) DESCR('TCP Cluster-rec
   ```
3. Define a sender channel from QM3 to QM1:
   ```
   DEFINE CHANNEL(DEMO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)') CLUSTER(DEMO) DESCR('TCP Cluster-sende
   ```
4. Verify the setup with the following commands:
   ```
   display topic(scores) type(all) clusinfo
   display clusqmgr(*)
   display chstatus(*)
   ```
5. Test the setup using 2 command windows:
   a. Enter this command in the first command window:
      ```
      /opt/mqm/samp/bin/amqspub /FOOTBALL/scores QM2
      ```
   b. Enter this command in the second command window:
      ```
      /opt/mqm/samp/bin/amqssub /FOOTBALL/scores QM3
      ```

Publish/subscribe clusters:

A publish/subscribe cluster is a standard IBM WebSphere MQ cluster of interconnected queue managers where publications are automatically moved from publishing applications to subscriptions that exist on any of the queue managers in the cluster.

A cluster that is used for publish/subscribe messaging is no different from a standard IBM WebSphere MQ cluster. As such, the queue managers within the publish/subscribe cluster can exist on physically separate computers and each pair of queue managers is automatically connected together by cluster channels when necessary. For information about how to plan and configure a IBM WebSphere MQ cluster, see How clusters work.

A publish/subscribe cluster is created when a clustered topic object is defined, through setting the **CLUSTER** attribute on the topic configured by any queue manager in the cluster. The topic definition is propagated to all members of the cluster. You can publish and subscribe to the topic, and any topic string below it in the topic tree, on any queue manager in the cluster. Publications are automatically propagated to subscribers connected to other queue managers in the cluster.

Non-clustered publish/subscribe activity can also take place in a publish/subscribe cluster, by working with topic strings which do not fall under a clustered topic object. This arrangement differs from a publish/subscribe hierarchy, in which all subscriptions are propagated across the whole hierarchy. In both cases, finer control is available using Subscription and Publication scope.

Using clusters in a publish/subscribe topology provides the following benefits:

- Messages destined for a subscription on a specific queue manager in the same cluster are transported directly to that queue manager and do not need to pass through an intermediate queue manager. This improves performance and optimizes inter-queue manager publish/subscribe traffic, in comparison with a hierarchical topology.
- Because all queue managers are directly connected to each other, there is no single point of failure in this topology. If one queue manager is not available, subscriptions on other queue managers in the cluster are still able to receive messages from publishers on available queue managers.
- In a system that contains multiple separate clusters, for example where the clusters are geographically dispersed, it is possible to connect clusters into a hierarchy of clusters. This connection is created by joining a single queue manager in each cluster to enable the flow of publications and subscriptions through the network; see "Combine the topic spaces of multiple clusters" on page 90. You can also control which publications flow from one cluster to another; see "Combine and isolate topic spaces in multiple clusters" on page 91 .
- A subscribing application can connect to its nearest queue manager, to improve its own performance. The queue manager receives all messages that match the subscription registration of the client from all queue managers within the cluster.

  The performance of a client application is also improved for other services that are requested from this queue manager. A client application can use both publish/subscribe and point-to-point messaging.
- The number of clients and subscriptions for each queue manager can be reduced by adding more queue managers to the cluster to share workload. Publications automatically distribute to the clients on the new queue managers. For some usage patterns, this process can make a publish/subscribe cluster topology highly scalable.

Things to consider when using clusters in publish/subscribe:

- All queue managers in a publish/subscribe cluster are automatically made aware of all other queue managers in the cluster. This process is different for a point-to-point cluster, where only queue managers of interest to a queue manager are known of.
- Queue managers in a publish/subscribe cluster that host one or more subscriptions to a clustered topic, automatically create cluster sender channels to all other queue managers in the cluster. The queue

managers also send information regarding the subscriptions to each of them, even when the receiving queue managers are not publishing messages on any clustered topics.

- The first subscription on a queue manager to a topic string under a clustered topic results in a message being sent to every other queue manager in the cluster. Similarly, the last subscription on a topic string to be deleted also results in a message. The more individual topic strings being used under a clustered topic, the more inter-queue manager communication occurs.

CAUTION:

For the reasons listed previously in this topic, the introduction of a clustered topic into a large IBM WebSphere MQ cluster (that is, a cluster that contains many queue managers) can immediately result in additional load on each queue manager in the cluster, and in some situations cause a reduction in performance. For more information, see “Cluster topic performance” on page 65.

Introducing publish/subscribe into a cluster of queue managers, especially an existing cluster, must be carefully planned to accommodate these reductions in performance.

Where it is known that a cluster cannot accommodate the reduction in performance of publish/subscribe, it is possible to disable the clustered publish/subscribe functionality in queue managers using the **PSCLUS** parameter. The **PSCLUS** parameter is primarily to stop the severe problems that can occur with the creation of a publish/subscribe cluster by accidentally or incorrectly defining a clustered topic. For more information about disabling this functionality, see “Inhibiting clustered publish/subscribe in a cluster” on page 64.

Publish/subscribe clustering: Best practices:

This topic provides guidance for planning and administering IBM WebSphere MQ Publish/subscribe clusters. The information is based on testing and feedback from customers.

The following information assumes that the user has a basic understanding of IBM WebSphere MQ clusters, Publish/subscribe, and is familiar with the topics in “Distributed publish/subscribe” on page 39. This information is not intended as a "one size fits all" solution, but is instead trying to share common approaches to common problems.

Publish/Subscribe clusters

With a cluster, you have "any-to-any" direct connectivity between queue managers in the cluster when required. When a cluster is used for point-to-point messaging, each queue manager in the cluster knows only the information about other cluster resources, such as other queue managers in the cluster and clustered queues, when applications connecting to them request to use them; that is they work on a need-to-know basis.

A publish/subscribe cluster is a cluster of queue managers, with the usual CLUSSDR and CLUSRCVR channel definitions. However, a publish/subscribe cluster also contains at least one TOPIC object that is defined on at least one queue manager in the cluster where the topic object has identified a cluster name.

With a topic object defined in the cluster, an application that is connected to one queue manager in the cluster can subscribe to that topic, or any node in the topic tree below that topic, and receive publications on that topic from other queue managers in the cluster. This process is achieved by the creation of proxy subscriptions on all the other queue managers in the cluster identifying the queue manager where the subscription exists. So, when a publication to the topic in question happens on their queue manager, they know to forward it to other appropriate members of the cluster, and from there deliver it to the individual application subscriptions.

To achieve this delivery, every queue manager in the cluster needs to know the identity of every other queue manager in the cluster, as soon as a topic is added to a cluster. This knowledge is propagated by way of the cluster's full repository queue managers. Published messages on one queue manager are only sent to other queue managers in the cluster that are known to host subscriptions to the same topic. To achieve this process, when an application creates a subscription to a topic that is clustered, that queue manager must communicate directly with every other queue manager in the cluster, by way of cluster-sender channels to propagate the proxy subscriptions.

This process differs greatly from the limited need-to-know information and connectivity that is needed when you are using a cluster for point-to-point delivery. Therefore, the requirements on a publish/subscribe cluster are different to the requirements on a point-to-point cluster (one without any clustered topics).

Using clustered topics makes extending the publish/subscribe domain between queue managers simple, but can lead to problems if the mechanics and implications are not understood, and considered in respect to the cluster being used for publish/subscribe. The following best practices are designed to help in this understanding and preparation.

In summary, the performance implications of clustered publish/subscribe can be detrimental to a large cluster, and needs to be carefully considered and understood before any attempt to use publish/subscribe in an existing cluster. For example, even the simple creation of a clustered topic object. It might be better to start with a small new cluster dedicated to publish/subscribe activity, and grow the cluster from there.

Designing a publish/subscribe topology

As previously described, there are capacity and performance considerations when you are using publish/subscribe in a cluster. Therefore, it is best practice to carefully consider the need for publish/subscribe across queue managers, and to limit it to only the number of queue managers that require it. After the minimal set of queue managers that need to publish and subscribe to a set of topics are identified, they can be made members of a cluster which contains only them and no other queue managers.

This is especially true in an established cluster already functioning well for point-to-point messaging. For this reason, when you are turning an existing large cluster into a publish/subscribe cluster, it is a better practice to initially create a separate cluster for the publish/subscribe work where the applications can be tried, rather than using the current cluster. It is possible to continue to use existing queue managers already in one or more point-to-point clusters, the subset of these queue managers need to be made members of the new publish/subscribe cluster. However, this new cluster must have separate queue managers that are configured as full repositories to isolate the additional load from the existing cluster full repositories.

Where you establish that a cluster is not to be used for publish/subscribe due to its size or current load, it is good practice to prevent this cluster unexpectedly being made into a publish/subscribe cluster by the simple creation of a clustered topic on any queue manager in the cluster. Use the **PSCLUS** queue manager property to achieve this design, for details, see Inhibiting clustered publish/subscribe in a cluster.

It is also important to choose carefully which topics are to be added to the cluster: The higher up the topic tree these topics are, the more widespread they become. For this reason, it is not recommended to put the topic root node into the cluster without considering the behavior that is seen. Make global topics obvious where possible, for example by using a high-level qualifier in the topic string: `/global` or `/cluster`.

How to size systems

Publish/subscribe clusters require many channels because the model is different to point-to-point messaging: There is the need for each queue manager to talk to all other queue managers in that cluster. The point-to-point model is an 'opt in' one, but publish/subscribe clusters have an indiscriminate nature with subscription fan-out. Therefore, the full repository queue managers, and any queue manager that hosts local subscriptions in a publish/subscribe cluster, must have the capacity to establish channels to every member of the cluster at the same time.

It is best to ensure that every queue manager in the publish/subscribe cluster can achieve this capacity, but it is acknowledged that queue managers that are known never to host subscriptions do not need to establish channels with every other queue manager, and therefore do not require this level of capacity.

However, care must be taken because an accidental subscription created on such a queue manager, or any attempt to manually resynchronize such a queue manager with the others in the cluster, results in all channels being started concurrently. See "Resynchronization of proxy subscriptions" for more information.

Clustered publish/subscribe enables the delivery of published messages on one queue manager to be delivered to subscriptions on other queue managers. But as for point-to-point messaging, the cost of transmitting messages between queue managers can be detrimental to performance. Therefore, attempts must be made to wherever possible to create subscriptions to topics on the same queue managers as where messages are being published.

Another consideration is the affect on performance on the system of propagating proxy subscriptions. Typically, a queue manager sends a proxy subscription message to every other queue manager in the cluster when the first subscription for a specific clustered topic string (not just a configured topic object) is created. If a publish/subscribe solution consists of many unique topic strings being subscribed to, or the topics are frequently being subscribed and unsubscribed from, a significant amount of proxy subscription traffic can be generated between all queue managers in a cluster, adversely affecting the overall performance of the system. See "Cluster topic performance" on page 65 for information about ways to reduce proxy subscription overhead.

Resynchronization of proxy subscriptions

Under normal circumstances, queue managers automatically ensure that the proxy subscriptions in the system correctly reflect the subscriptions on each queue manager in the cluster.

However, should the need arise, you can manually resynchronize a queue manager's local subscriptions with the proxy subscriptions that it propagated across the cluster using the REFRESH QMGR TYPE(PROXYSUB) command.

Note: Resynchronization temporarily creates a sudden additional proxy subscription load on the cluster, originating from the queue manager where the command is issued. For this reason, do not use it unless IBM WebSphere MQ service, IBM WebSphere MQ documentation, or error logging instructs you to do so.

An example of when resynchronization is required is when a queue manager cannot correctly propagate its proxy subscriptions, perhaps because a channel has stopped and all messages cannot be queued for transmission, or because operator error has caused messages to be incorrectly deleted from the `SYSTEM.CLUSTER.TRANSMIT.QUEUE` queue. In this situation, first rectify the original problem (for example by restarting the channel) then issue the **`REFRESH QMGR TYPE(PROXYSUB)`** command on the queue manager. Note that publications missed due to proxy subscriptions not being in place are not recovered for the affected subscriptions. This drawback must be taken into account.

Resynchronization requires the queue manager to start channels to all other queue managers in the cluster. Therefore the queue manager that you are refreshing must have enough capability to cope with communicating with every other queue manager in the cluster.

Cluster topics:

Cluster topics are administrative topics with the **cluster** attribute defined. Information about cluster topics is pushed to all members of a cluster and combined with local topics to create a different topic space at each queue manager.

When you define a cluster topic on a queue manager, the cluster topic definition is sent to the full repository queue managers. The full repositories then propagate the cluster topic definition to all queue managers within the cluster, making the same cluster topic available to publishers and subscribers at any queue manager in the cluster. The queue manager on which you create a cluster topic is known as a cluster topic host. The cluster topic can be used by any queue manager in the cluster, but any modifications to a cluster topic must be made on the queue manager where that topic is defined (the host) at which point the modification is propagated to all members of the cluster via the full repositories.

At each queue manager a single topic name space is constructed from the local and cluster topic definitions that it is aware of. When an application subscribes to a topic that resolves to a clustered topic, IBM WebSphere MQ creates a proxy subscription and sends it directly from the queue manager where the subscription is created, to all the other members of the cluster. Unlike the clustered topic itself, proxy subscriptions do not flow via the full repository queue managers.

Messages published on a topic are sent to every subscription known to the queue manager that the publisher is connected to. If any of those subscriptions are proxy subscriptions, a copy of the published message is sent to the queue manager that originated the proxy subscription. The receiving queue manager then sends a copy of the message to every local subscription. This process ensures that the subscriber to a clustered topic receives publications from publishers connected to any of the queue managers in the cluster and that the minimal number of published messages are propagated through the cluster.

If you have a clustered topic, and a local topic object, then the local topic takes precedence. See “Multiple cluster topic definitions” on page 63 for more information.

For more information about the commands to use to display cluster topics, see the following related links:

Wildcard subscriptions

Proxy subscriptions are created when local subscriptions are made to a topic string that resolves at, or below, a clustered topic object. If a wildcard subscription is made higher in the topic hierarchy than any cluster topic, it does not have proxy subscriptions sent around the cluster for the matching cluster topic, and therefore receives no publications from other members of the cluster. It does however receive publications from the local queue manager.

However, if another application subscribes to a topic string that resolves to or below the cluster topic, proxy subscriptions are generated and publications are propagated to this queue manager. On arrival the original, higher wildcard subscription is considered a legitimate recipient of those publications and receives a copy.

This behavior differs from locally published messages on the same topics. If this behavior is not required, setting **WILDCARD(BLOCK)** on the clustered topic makes the original wildcard not be considered a legitimate subscription and does not receive any publications (local, or from elsewhere in the cluster) on the cluster topic, or its subtopics.

Related information:
Working with administrative topics
Working with subscriptions
DISPLAY TOPIC
DISPLAY TPSTATUS
DISPLAY SUB

Cluster topic attributes:

A good understanding of cluster topic attributes is needed to design and administer publish/subscribe clusters.

A topic object has a number of attributes that apply to multi-queue manager publish/subscribe topologies. When you are using an IBM WebSphere MQ cluster to create such a topology, these attributes have the following behavior.

PROXYSUB

- **PROXYSUB** is an attribute that controls when proxy subscriptions are made. For details about why you might want to change this attribute from the default value of FIRSTUSE, see "More on routing mechanisms" on page 44.
- In the same way as for other attributes of a clustered topic, the **PROXYSUB** attribute is propagated to every queue manager in the cluster, not just the queue manager that the topic was defined on. This instantly results in every queue manager in the cluster creating a wildcarded proxy subscription to every other queue manager. The result of this process is that every queue manager creates cluster sender channels to every other queue manager, and any messages that are published are sent to every queue manager.

PUBSCOBE and SUBSCOPE

PUBSCOPE and **SUBSCOPE** determine whether this queue manager propagates publications to queue managers in the topology (publish/subscribe cluster or hierarchy) or restricts the scope to just its local queue manager. You can do the equivalent job programmatically using MQPMO_SCOPE_QMGR / MQSO_SCOPE_QMGR.

- **PUBSCOPE** If a cluster topic object is defined with **PUBSCOPE(QMGR)**, the definition is shared with the cluster, but the scope of publications that are based on that topic is local only and they are not sent to other queue managers in the cluster.
- **SUBSCOPE** If a cluster topic object is defined with **SUBSCOPE(QMGR)**, the definition is shared with the cluster, but the scope of subscriptions that are based on that topic is local only, therefore no proxy subscriptions are sent to other queue managers in the cluster.

These two attributes are commonly used together to isolate a queue manager from interacting with other members of the cluster on particular topics. The queue manager neither publishes or receives publications on those topics to and from other members of the cluster. This situation does not prevent publication or subscription if topic objects are defined on subtopics.

Setting **SUBSCOPE** to QMGR on a local definition of a topic does not prevent other queue managers in the cluster from propagating their proxy subscriptions to the queue manager if they are using a clustered version of the topic, with **SUBSCOPE(ALL)**. However, if the local definition also sets **PUBSCOPE** to QMGR those proxy subscriptions are not sent publications from this queue manager.

Multiple cluster topic definitions:

A local topic definition overrides a remotely defined cluster topic definition of the same name. Creation of multiple definitions of the same cluster topic on different queue managers in a cluster is also possible. Both of these scenarios require some caution however, the reasons are explained in this topic.

Just as for clustered queues, having multiple definitions of the same cluster topic object in a cluster introduces the possibility of different properties defined on each. It is not easy to determine which version of the topic definition is seen by each queue manager in the cluster and it is therefore hard to determine the expected behavior.

Where two or more cluster topic definitions, for a single topic string, have differing attributes or exist in more than one cluster, messages (AMQ5465 & AMQ5466) are written to the error log and the most recently received cluster topic definition is used.

The cluster topic host queue manager must not delete the topic definition, and remains in the cluster to ensure that the clustered topic continues to be known by all members of the cluster. It is not essential that this host queue manager is continually available because the cluster topic definition is cached by the full repository queue managers and by all other queue managers in their partial cluster repositories. This caching allows for at least 60 days of availability while the host queue manager is unavailable. For more information about this subject, see "Key roles for publish/subscribe cluster queue managers" on page 69.

Overriding a cluster topic definition locally

It might be necessary to override the behavior of a clustered topic on certain queue managers in the cluster. This override can be achieved by defining a local topic object to override a cluster topic object with the same topic string, and use it to publish only to locally connected subscribers.

Even when a local definition of a topic is created to override a clustered topic on a queue manager the queue manager continue to receive proxy subscriptions from other members of the cluster using the clustered topic definition. By default, messages published locally continue to be sent to the remote queue managers to honor the proxy subscriptions. If this arrangement is not required, specify **PUBSCOPE**(QMGR) on the local topic object to ensure publisher applications connected to this queue manager publish only to local subscribers.

Modifying a cluster topic definition

If you need to alter a cluster topic definition, modify it at the same queue manager it was defined on, the cluster topic host. Do not create a definition of the same cluster topic on a different queue manager in the cluster. Defining the topic again results in two cluster topic hosts for the same cluster topic.

Defining a cluster topic multiple times creates potentially conflicting definitions and the possibility that different queue managers use different definitions at different times.

Moving a cluster topic definition to a different queue manager in the cluster

You might need to move a cluster topic definition from one queue manager in the cluster to another, for example when decommissioning a queue manager from the cluster. To move a cluster topic definition to a different queue manager in the cluster without interrupting the flow of publications, you need to follow these steps. The example moves a definition from QM1 to QM2.

1. Create a duplicate of the cluster topic definition on QM2 with the same attributes as the definition of QM1.
2. Wait for the new definition to be propagated throughout the cluster by the full repository queue managers. The propagation can be determined by displaying the cluster topics on each cluster member using the **(DISPLAY CLUSTER)** command, and checking for a definition originating from QM2.

3. Delete the cluster topic definition from QM1.

After the original definition is deleted from QM1 it is possible to modify the definition on QM2 if required, without introducing a conflict in properties.

Replacing a cluster topic definition on a failed queue manager

In the previous scenario, it might not be possible to delete the definition from QM1 if QM1 is not available for a period of time. In this scenario it is acceptable to run with both definitions in existence.

If it then becomes a requirement to modify the clustered topic definition, it is possible to modify the version on QM2 in the knowledge that the QM2 definition is newer than the QM1 definition, and therefore prevails. However, during this period, errors are written to the error logs of the queue managers because there is a conflicting cluster topic definition. Resolve the error as soon as possible by removing the duplicate cluster topic definition from QM1 when it can be restarted.

Alternatively, if QM1 is never going to return to the cluster (for example, unexpected decommissioning following a catastrophic hardware failure), the RESET CLUSTER command can be used to forcibly eject the queue manager. **RESET CLUSTER** automatically deletes all topic objects hosted on the target queue manager.

Inhibiting clustered publish/subscribe in a cluster:

Introducing publish/subscribe into a cluster of queue managers, especially an existing cluster, must be carefully planned to accommodate any reductions in performance.

The introduction of a clustered topic into a large IBM WebSphere MQ cluster (one that contains many queue managers) can immediately result in additional load on each queue manager in the cluster, and in some situations, a reduction in performance. Therefore the introduction of publish/subscribe must be carefully planned. See "Cluster topic performance" on page 65 for more information.

Where it is known that a cluster could not accommodate the overheads of publish/subscribe it is possible to disable the clustered publish/subscribe functionality in queue managers by setting the queue manager attribute **PSCLUS** to DISABLED.

Setting **PSCLUS** to DISABLED modifies three aspects of queue manager functionality:

- An administrator of this queue manager is no longer able to define a Topic object as clustered.
- Incoming topic definitions or proxy subscriptions from other queue managers are rejected (a warning message is logged to inform the administrator of incorrect configuration).
- Full repositories no longer share information about every queue manager with all other partial repositories automatically when they receive a topic definition.

Although **PSCLUS** is a parameter of each individual queue manager in a cluster, it is not intended to selectively disable publish/subscribe in a subset of queue managers in the cluster. Apart from anything else, this method would cause frequent error messages to be seen as proxy subscriptions and topic definitions would constantly be seen and rejected. Ideally, when using this option, consistently set all queue managers in the cluster to disabled. Where a queue manager participates in one or more publish subscribe cluster or clusters, and also one or more traditional cluster or clusters, **PSCLUS** must be set to ENABLED on that queue manager. See the following information about disabling at the full repositories.

Importantly, setting **PSCLUS** to DISABLED on all full repository queue managers in the cluster prevents any clustered topic definition on an incorrectly configured partial repository from affecting other queue managers in the cluster. In such scenarios, the inconsistency is reported in the error logs of the full repository queue managers.

When overlapping a traditional point to point cluster with a publish subscribe cluster, it is important to use a separate set of full repositories in each. This arrangement allows topic definitions and 'all queue manager' information to be allowed to flow only in the publish/subscribe cluster.

There are some caveats on usage of this parameter which help to avoid inconsistent configurations. No clustered topic objects can exist in any cluster of which this queue manager is a member when modifying from `ENABLED` to `DISABLED`. Any such topics (even remotely defined ones) must be deleted before disabling this function

For more information about **`PSCLUS`**, see ALTER QMGR (PSCLUS).

Cluster topic performance:

The performance characteristics of cluster topics require special consideration because they differ from the performance characteristics of cluster queues, and poorly considered usage can potentially be a source of performance problems in large or unbalanced clusters.

Reducing the effect of publish/subscribe on performance

There are two sources of workload on a queue manager in a cluster: directly handling messages for application programs, and handling messages and channels needed to manage the cluster. In a typical point-to-point cluster, the cluster system workload is largely limited to information explicitly requested by members of the cluster as required (see a comparison in "Performance characteristics of publish/subscribe clusters" on page 68). Therefore in anything other than a very large cluster, for example one which contains thousands of queue managers, you can largely discount the performance effect of managing the cluster when considering queue manager performance.

In a publish/subscribe cluster, information, such as clustered topics and proxy subscriptions, is pushed to all members of a cluster, irrespective of whether all cluster queue managers are actively participating in publish/subscribe messaging. This process can create a significant additional load on the system. Therefore you do need to consider the effect of cluster management on queue manager performance, both in its timing, and its size.

To reduce the effect of publish/subscribe cluster management on the performance of a cluster consider the following two suggestions:

1. Perform cluster, topic, and subscription updates at off-peak times of the day.
2. If you are considering adding publish/subscribe topics to an existing large cluster just because the cluster is already there, consider whether you can define a much smaller subset of queue managers involved in publish/subscribe and make that an "overlapping" cluster. This cluster is then the cluster where cluster topics are defined. Although some queue managers are now in two clusters, the overall effect of publish/subscribe is reduced:
   a. The size of the publish/subscribe cluster is smaller.
   b. Queue managers not in the publish/subscribe cluster are much less affected by the effect of cluster management traffic.

Balancing producers and consumers

An important concept in asynchronous messaging performance is *balance*. Unless message consumers are balanced with message producers, there is the danger that a backlog of unconsumed messages might build up and seriously affect the performance of multiple applications.

In a point-to-point messaging topology, the relationship between message consumers and message producers is readily understood. You can obtain estimates of message production and consumption, queue by queue, channel by channel. If there is a lack of balance, the bottlenecks are readily identified and then remedied.

It is harder to work out whether publishers and subscribers are balanced in a publish/subscribe topology. Start from each subscription that resolves to a clustered topic, and work back to the queue managers having publishers on the topic. Calculate the number of publications flowing to each subscriber from each queue manager.

Each publication that matches a subscription on a remote queue manager in the cluster (based on proxy subscriptions) is put to the `SYSTEM.CLUSTER.TRANSMIT.QUEUE` . If multiple remote queue managers have proxy subscriptions for that publication, multiple copies of the message are put to the transmission queue, targeted for different cluster sender channels.

Those publications are targeted at the `SYSTEM.INTER.QMGR.PUBS` queue on the remote queue managers. Each queue manager processes messages arriving on that queue and deliver them to the correct subscriptions on that queue manager.

For this reason, monitor the load at the following points where bottlenecks might arise:

- The individual subscription queues themselves:
  - This bottleneck would imply that the subscribing application is not consuming the publications as quick as they are being published.
- The `SYSTEM.INTER.QMGR.PUBS` queue:
  - The queue manager is receiving publications from one or more remote queue managers faster than it can distribute them to the local subscriptions.
- The cluster channels between the publishing queue manager, the subscribing queue managers, and the cluster transmission queues .(`SYSTEM.CLUSTER.TRANSMIT.QUEUE` by default) on the publishing queue manager:
  - Either one or more cluster channel is not running, or messages are being published to the local queue manager faster than the channels can deliver them to the remote queue manager.
- If the publishing application is using a queued publish/subscribe interface, the `SYSTEM.BROKER.DEFAULT.STREAM` queue and any other stream queues listed in the `SYSTEM.QPUBSUB.QUEUE.NAMELIST` , and the `SYSTEM.BROKER.DEFAULT.SUBPOINT` queue and any other subpoint queues, as listed in the `SYSTEM.QPUBSUB.SUBPOINT.NAMELIST`, also require consideration:
  - Messages are being put by local publishing applications faster than the local queue manager can process the messages.

Subscription performance considerations

As previously described, when a subscription is made on a queue manager for a topic string that resolves to a clustered topic, that queue manager must ensure that every other queue manager in the cluster has a proxy subscription in place for the topic. To achieve this result, the queue manager creates and sends a proxy subscription message to every other queue manager in the cluster.

Using the default configuration the only time that creating a subscription to a clustered topic does not result in new proxy subscriptions being sent, is when there is already a subscription on the local queue manager to exactly the same topic string. In this situation, no additional proxy subscriptions are required because arriving publications are delivered to all matching subscriptions, not just the original subscription for the topic string.

For an alternative to the default configuration, see “Disabling individual proxy subscriptions” on page 67.

Subscription selectors are not taken into account, so two subscriptions to the same topic string, but with different selectors, still share proxy subscriptions. This situation can also mean that publications that match the topic string are propagated to the subscriber queue manager, even if the publication does not match the subscription's selector.

An equivalent message to the proxy subscription message is created and sent to all queue managers when the last subscription to a topic string is deleted from a queue manager. This process removes the proxy subscriptions from the remote queue managers.

For these reasons, the size of the cluster and frequency of subscriptions to different topic strings can exert a significant load on the cluster itself and must be considered when planning the cluster and the topics to be used by the publish/subscribe applications.

When considering the load on the system from the proxy subscription traffic, in addition to monitoring the queues listed in the “Balancing producers and consumers” on page 65 section, monitor the following queues.

- The SYSTEM.INTER.QMGR.FANREQ queue on the subscriber queue manager.
- The SYSTEM.INTER.QMGR.CONTROL queue on all other queue managers in the cluster.

Any significant message backlog on these queues implies that either the rate of subscription change is too great for the system or a queue manager is not correctly functioning in the cluster. Either due to having publish/subscribe support disabled (see **PSMODE** in ALTER QMGR) or a problem occurring that requires further investigation, at which point check the queue manager error logs.

Reducing proxy subscription traffic

If proxy subscription overhead is high, steps should be taken to reduce this. This might be possible through general topic consolidation or through changing to a broadcast model for inter-queue manager publications.

It is a general publish/subscribe recommendation that the use of topic strings is assessed to see whether they can be consolidated in a way to reduce the overall load on resources for the system. The use of many distinct, transient, topic strings introduces some level of management overhead on each queue manager in the system where publishers or subscriptions are attached. Reducing the number and transient nature of topic strings and therefore the publishers and subscriptions to them reduces the impact on the system.

One method for reducing proxy subscription traffic is to locate subscriptions to the same topic string on the same queue manager. This method allows this queue manager to send a single proxy subscription to the other queue managers, rather than having multiple queue managers sending proxy subscriptions, each for their own set of subscriptions on the same topic string. This practice also optimizes publication routing across the cluster.

Disabling individual proxy subscriptions

In some situations, where the set of distinct topic strings being subscribed to across a cluster is large and continually changing, it might be better to change from a subscription propagation model to a publication broadcast model. This preferred model is where every publication on any cluster topic is automatically sent to every queue manager in the cluster, irrespective of the existence of subscriptions on those queue managers.

The receiving queue managers can then deliver the messages to the local subscriptions that exist, or discard the message. In this model, there is no need for individual proxy subscriptions to be created and deleted based on the existence of subscriptions. When running in this mode it is probable that the published message resource load increases as all publications are sent to all queue managers. Therefore, the queue managers in the cluster must have the capacity to handle this additional load.

Enable a broadcast model using the following configuration steps:

1. Every queue manager hosting subscriptions must be configured to not send proxy subscriptions that match local subscriptions to clustered topics. This configuration requires the following tuning parameter to be set in each queue manager `qm.ini` file, prior to the definition of cluster topics or the creation of subscriptions in the cluster:

   ```
   TuningParameters:
    pscProxySubFlags=1
   ```
2. After the tuning parameter is set, all queue managers must be restarted.
3. After the queue managers are restarted the clustered topic/topics can be defined. Each cluster topic must set **PROXYSUB** to `FORCE`.

Reversing the behavior

To reverse the mode of operation described previously in “Disabling individual proxy subscriptions” on page 67, use the following steps:

1. Remove the tuning parameter from the `qm.ini` file for every queue manager.
2. Restart every queue manager.
3. Issue the **REFRESH QMGR TYPE(PROXYSUB)** command on every queue manager hosting subscriptions.
4. Set **PROXYSUB** to `FIRSTUSE` on the clustered topic or topics.

CAUTION: In both the enabling and reversing of this behavior, if all the steps are not completed in the documented order, correct flow of publications to subscriptions might not occur.

Note: Implication of setting PROXYSUB (to FORCE)

As described previously in this topic, the **PROXYSUB(FORCE)** topic attribute can reduce the proxy subscription traffic, but it must be used with caution. The **PROXYSUB(FORCE)** attribute is propagated to every queue manager in the cluster, not just the queue manager that the topic was defined on. This instantly results in every queue manager in the cluster creating a wildcarded proxy subscription to every other queue manager. The result of this process is that every queue manager creates cluster sender channels to every other queue manager, and any published messages are sent to every queue manager.

Setting this property in a large or busy cluster can result in additional load on system resources.

Performance characteristics of publish/subscribe clusters

It is important to consider how changing attributes of a publish/subscribe cluster, such as adding a queue manager, topic, or subscription to the cluster affects the performance of applications running in the cluster.

Compare a point-to-point cluster with a publish/subscribe cluster in respect of two management tasks.

First, a point to point cluster:

1. When a new cluster queue is defined, the destination information is pushed to the full repository queue managers, and only sent to other cluster members when they first reference a cluster queue (for example, an application attempts to open it). This information is then cached locally by the queue manager to remove the need to remotely retrieve the information each time the queue is accessed.
2. Adding a queue manager to a cluster does not directly affect the load on other queue managers. Information about the new queue manager is pushed to the full repositories, but channels to the new queue manager from other queue managers in the cluster are only created and started when traffic begins to flow to or from the new queue manager.

In short, the load on a queue manager in a point-to-point cluster is related to the message traffic it handles for application programs and is not directly related to the size of the cluster.

Second, a publish/subscribe cluster:

1. When a new cluster topic is defined, the information is pushed to the full repository queue managers, and from there directly to all members of the cluster immediately, causing channels to be started to each member of the cluster from the full repositories if not already started.
2. When a subscription is created to a cluster topic on a new topic string, the information is pushed directly from that queue manager to all other members of the cluster immediately, causing channels to be started to each member of the cluster from that queue manager if not already started.
3. When a new queue manager joins an existing cluster, information about all clustered topics is pushed to it from the full repository queue managers. The new queue manager then synchronizes knowledge of all subscriptions to cluster topics in the cluster with all members of the cluster, causing channels to be created and started to each member of the cluster from the new queue manager.

In summary, cluster management load at any queue manager in the cluster grows with the number of queue managers, clustered topics, and proxy subscriptions within the cluster, irrespective of the local use of those cluster topics on each queue manager.

Key roles for publish/subscribe cluster queue managers:

Similar to point-to-point clusters, there are two key roles for queue managers in a publish/subscribe cluster; as full repository queue managers and as cluster topic hosts.

Full repository

A full repository queue manager has the role of pushing object definitions out to other members of a cluster; in the case of publish/subscribe clusters, pushing clustered topic object definitions out to other members of the cluster.

Cluster topic host

A cluster topic host is a queue manager where a clustered topic object is defined. You can define clustered topic objects on any queue manager in the publish/subscribe cluster. The cluster topic object is pushed to the full repository queue managers, which then push it out to all the other queue managers in the cluster where it is cached for use by publishers and subscribers running on any queue managers in the cluster.

Availability and management

You should define two full repositories in a cluster to maximize the availability of cluster topic definitions in the cluster.

As for queued messaging clusters, in publish/subscribe clusters that have just two highly available computers among many computers, it is good practice to define the highly available computers as full repositories.

In queued clusters, you can increase the availability and throughput of a cluster queue by *defining* the same cluster queue on multiple queue managers in the cluster. Messages are then workload balanced across them. In contrast, in publish/subscribe clusters, a clustered topic is *available* on all queue managers in the cluster but no workload balancing of publish/subscribe traffic is performed. Instead, separate subscriptions and publishers should be spread across different queue managers, to spread the publish/subscribe load. If the queue manager on which you defined the cluster topic becomes unavailable, the other queue managers continue to process publish/subscribe requests for the topic.

However, if the queue manager on which you defined the cluster topic object is never made available again, then eventually the cached topic objects on the other queue managers are deleted and the topic becomes unavailable. This process happens after at least 60 days (depending on when the topic definition was last refreshed) from when the topic definition became unavailable.

With the 60 day period to recover the queue manager on which you defined cluster topic objects, there is little need to take special measures to make a cluster topic host highly available. The 60 day period is sufficient to cater for technical problems; the 60 day period is likely to be exceeded only because of administrative errors. To mitigate that possibility, if the cluster topic host is unavailable, all members of the cluster write error log messages hourly that their cached cluster topic object was not refreshed. Respond to this message by making sure that the queue manager on which the cluster topic object is defined, is running.

You might adopt the practice of defining the same cluster topic object on other queue managers. Each definition results in an additional cluster topic object being pushed out to the other queue managers in the cluster, including the other cluster topic hosts. Now if a cluster topic host becomes unavailable for over 60 days, only its version of the cluster topic object is removed from the other hosts. The other versions of the cluster topic object remain. It is a requirement that all definitions for a specific topic in a cluster are identical, otherwise it is difficult to ascertain which topic definition is being used by a queue manager. The most recent copy on any host is always the cluster topic object that is used.

Weigh the added protection of multiple cluster topic definitions against increased administrative complexity: with increased complexity comes a greater chance of human error.

Unlike hosting a clustered queue, being the host queue manager for a clustered topic definition does not introduce any additional application message traffic. That traffic is limited to the queue managers where the subscriptions are created and the messages published. It is possible to host the clustered topic on a queue manager that is doing neither. This situation means that although it is not mandatory, it is often sensible to host the clustered topics on the full repository queue managers for the cluster as these queue managers might be provisioned with higher levels of availability, and have tighter administrative control on them. This arrangement reduces the possibility of incorrectly modifying or deleting the definitions or even the queue manager.

Overlapping cluster support, and publish/subscribe:

With IBM WebSphere MQ clusters, a single queue manager can be a member of more than one cluster. This arrangement is known as overlapping clusters. Clustered topics in a publish/subscribe clusters behave differently to queues when clusters are overlapped in a queue manager. This behavior must be clearly understood when using clustered publish/subscribe with overlapping clusters.

Unlike for a queue, there is no ability to associate a topic definition with more than one cluster. Therefore, the scope of proxy subscriptions created in a cluster is limited to the single cluster in which the clustered topic is defined. However, each queue manager has a single topic tree which includes all local topics and any known clustered topics, from any cluster that they are a member of. For this reason it is possible to architect such a system where publish/subscribe behaviour can be hard to understand.

Integrating multiple publish/subscribe clusters

For point-to-point messages, a reason for making a single queue manager a member of more than one cluster is to create a cluster gateway between two clusters. For more information about this subject, see Overlapping clusters. This cluster gateway enables point-to-point messages originating in one cluster to be routed to queries in another cluster. Publish/subscribe clusters inherit the capability of being overlapped from traditional queue manager clusters. However, you cannot use this mechanism to route publications and subscriptions from one cluster to another.

Instead, to pass publications and subscriptions from queue managers in one cluster to another, you must link the queue managers together using a publish/subscribe hierarchy. This arrangement can be achieved by explicitly creating a parent-child hierarchical relationship between one queue manager in one cluster with another queue manager in the other cluster. This relationship enables the flow of all proxy subscriptions between the clusters, and thus any matching publications. For more information about this relationship, see “Publish/subscribe hierarchies” on page 73.

A way to limit which publications and subscriptions flow between clusters is to use a gateway queue manager that is in neither cluster; see “Combine and isolate topic spaces in multiple clusters” on page 91.

Overlapping clusters, single topic tree

Each queue manager has a single Topic tree that includes local topics and all known clustered topics. A further consideration with overlapping two clusters, both using publish/subscribe, is that it is possible for a queue manager in each cluster to define a clustered topic with the same name, or to define differently named clustered topics which have the same topic string. On the queue managers that are members of both clusters, conflicts arise when they are informed of the multiple cluster topic definitions, one for each cluster. A problem is reported by the queue manager but the queue manager continues to operate, using only the most recent clustered topic definition. Therefore the behavior becomes nondeterministic and cannot be relied upon.

For this reason, overlapping clusters that are using clustered publish/subscribe must consider their topic definition namespace to span all clusters, and name their topic objects and structure their topic strings accordingly. You can then use queue managers in the overlap to publish and subscribe to both clusters predictably.

In Figure 33, T_B and T_C are topic definitions that do not overlap. A publisher connected to QM3, in the cluster overlap, is able to publish to both topics in their respective clusters. A subscriber connected to QM3 in the overlap is able to subscribe to topics in both clusters.

An alternative way of thinking about Figure 33 is to consider the proxy subscriptions. An application connected to queue manager QM3, subscribing on a topic that resolves to topic object T_B (which exists only in CLUSTER 1) results in proxy subscriptions being sent from queue manager QM3 to queue managers QM1 and QM2 only. An application connected to queue manager QM3 subscribes on a topic that resolves to topic object T_C (which exists only in the CLUSTER 2). The subscription results in proxy subscriptions being sent from queue manager QM3 to queue managers QM4 and QM5 only.

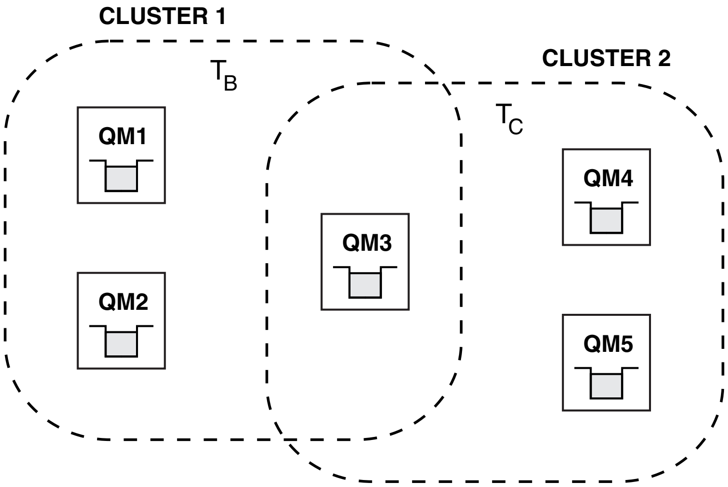


Figure 33. Overlapping clusters: two clusters each subscribing to different topics

Publishers and subscribers to queue managers that are not in the overlap can publish and subscribe to topics in their cluster only, for example a subscriber to a topic string on QM2 do not receive messages published to the same topic string published from QM5, irrespective of clustering the topics. To achieve this arrangement, a publish/subscribe hierarchy is required.

Overlapping clusters, wildcarded subscriptions

For the reasons in the previous section of this topic, care must be taken when using wildcards to subscribe to topics on a queue manager that is a member of multiple clusters.

In the previous example, assume that the two topic objects were configured as:

- T_B: Topic name `'Football'`, cluster `'CLUSTER1'.` topic string `'/Sport/Football'`
- T_C: Topic name `'Tennis'`, cluster `'CLUSTER2'.` topic string `'/Sport/Tennis'`

In this scenario, the two clustered topics are clearly separate, with no overlap in either the topic name or the topic string.

An application connected to `QM3` can create a subscription to `'/Sport/Football'` and a subscription to `'/Sport/Tennis'`. They would then receive any publications from across the two clusters. However, as described in “Administrative topic objects” on page 25, if they were to subscribe to `'/Sport/#'`, with the intent of receiving publications on both `'/Sport/Football'` and `'/Sport/Tennis'`, this model is not recognized as a clustered topic in either cluster and therefore no proxy subscriptions would be created. They would then miss publications from other queue managers in either cluster.

As already described, it is not valid to create a clustered topic for `'/Sport/#'` in both `CLUSTER 1` and `CLUSTER 2`, because these clustered topics would conflict and informational messages are written to the error logs to indicate this. However, it is 'allowed' to create such a topic in just one of the clusters, say `CLUSTER 1`. Now a subscription to `'/Sport/#'` in `QM3` would result in proxy subscriptions being sent to the queue managers in CLUSTER 1 only, so still, publications to `'/Sport/Tennis'` from `QM4` or `QM5` would still fail to be received.

The only solution in this scenario is to continue to create two separate subscriptions.

REFRESH CLUSTER considerations for publish/subscribe clusters:

Issuing the **REFRESH CLUSTER** command results in the queue manager temporarily discarding locally held information about a cluster, including any cluster topics and their associated proxy subscriptions.

The time taken from issuing the **REFRESH CLUSTER** command to the point that the queue manager regains a full knowledge of the necessary information for clustered publish/subscribe depends on the size of the cluster, the availability, and the responsiveness of the full repository queue managers.

During the refresh processing, disruption to publish/subscribe traffic in a publish/subscribe cluster occurs. For large clusters, use of the **REFRESH CLUSTER** command can disrupt the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster. For these reasons, the **REFRESH CLUSTER** command must be used in a publish/subscribe cluster only when under the guidance of your IBM Support Center.

The disruption to the cluster can appear externally as the following symptoms:

- Subscriptions to cluster topics on this queue manager are not receiving publications from publishers that are connected to other queue managers in the cluster.
- Messages that are published to cluster topics on this queue manager are not being propagated to subscriptions on other queue managers.
- Subscriptions to cluster topics on this queue manager created during this period are not consistently sending proxy subscriptions to other members of the cluster.
- Subscriptions to cluster topics on this queue manager deleted during this period are not consistently removing proxy subscriptions from other members of the cluster.
- 10-second pauses, or longer, in message delivery.
- **MQPUT** failures, for example, MQRC_PUBLICATION_FAILURE.
- Publications placed on the dead-letter queue with a reason of MQRC_UNKNOWN_REMOTE_Q_MGR

For these reasons publish/subscribe applications need to be quiesced before issuing the **REFRESH CLUSTER** command.

See also Usage Notes for **REFRESH CLUSTER** and Clustering: Using REFRESH CLUSTER best practices.

After a **REFRESH CLUSTER** command is issued on a queue manager in a publish/subscribe cluster, wait until all cluster queue managers and cluster topics have been successfully refreshed, then resynchronize proxy subscriptions as described in "Resynchronization of proxy subscriptions" on page 60. This arrangement requires cluster sender channels to be started from this queue manager to all other queue managers in the cluster. When all proxy subscriptions have been correctly resynchronized, restart your publish/subscribe applications.

If a **REFRESH CLUSTER** command is taking a long time to complete, monitor it by looking at the CURDEPTH of `SYSTEM.CLUSTER.COMMAND.QUEUE` .

Related information:

Clustering: Using REFRESH CLUSTER best practices
You use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster. You should not need to use this command, except in exceptional circumstances. If you do need to use it, there are special considerations about how you use it. This information is a guide based on testing and feedback from customers.

Publish/subscribe hierarchies:

Queue managers can be grouped together in a hierarchy, where the hierarchy contains one or more queue managers that are directly connected. Queue managers are connected together using a connection-time parent and child relationship. When two queue managers are connected together for the first time, the child queue manager is connected to the parent queue manager.

When the parent and child queue managers are connected in a hierarchy there is no functional difference between them until you disconnect queue managers from the hierarchy.

Note: IBM WebSphere MQ hierarchical connections require that the queue manager attribute PSMODE is set to ENABLED.

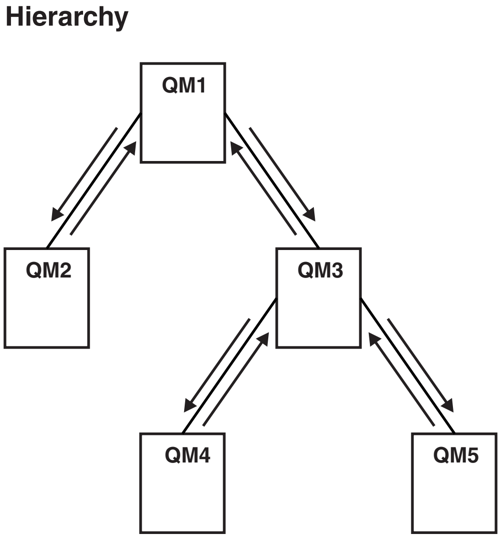


Figure 34. Simple publish/subscribe hierarchy

Connect a queue manager to a broker hierarchy:

You can connect a local queue manager to a parent queue manager to modify a broker hierarchy.

Before you begin

1. Enable queued publish/subscribe mode. See Starting queued publish/subscribe.
2. This change is propagated to the parent queue manager using a IBM WebSphere MQ connection. There are two ways to establish the connection.
   - Connect the queue managers to a IBM WebSphere MQ cluster, see Adding a queue manager to a cluster
   - Establish a point-to-point channel connection using a transmission queue, or queue manager alias, with the same name as the parent queue manager. For more information about how to establish a point-to-point channel connection, see WebSphere MQ distributed-messaging techniques.

About this task

Use the `ALTER QMGR PARENT` *(PARENT_NAME)* runmqsc command to connect children to parents.

Distributed publish/subscribe is implemented by using queue manager clusters and clustered topic definitions. For interoperability with IBM WebSphere MQ Version 6.0 and WebSphere Message Broker Version 6.1 and WebSphere Event Broker Version 6.1 and earlier, you can also connect Version 7.1 or later queue managers to a broker hierarchy as long as queued publish/subscribe mode is enabled.

Procedure

ALTER QMGR PARENT(PARENT)

Example

The first example shows how to attach QM2 as a child of QM1, and then querying QM2 for its connection:

```
C:>runmqsc QM2
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM2
alter qmgr parent(QM1)
     1 : alter qmgr parent(QM1)
AMQ8005: WebSphere MQ queue manager changed.
display pubsub all
     2 : display pubsub all
AMQ8723: Display pub/sub status details.
   QMNAME(QM2)                           TYPE(LOCAL)
   STATUS(ACTIVE)
AMQ8723: Display pub/sub status details.
   QMNAME(QM1)                           TYPE(PARENT)
   STATUS(ACTIVE)
```

The next example shows the result of querying QM1 for its connections:

```
C:\Documents and Settings\Admin>runmqsc QM1
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.
display pubsub all
     2 : display pubsub all
AMQ8723: Display pub/sub status details.
   QMNAME(QM1)                           TYPE(LOCAL)
   STATUS(ACTIVE)
AMQ8723: Display pub/sub status details.
   QMNAME(QM2)                           TYPE(CHILD)
   STATUS(ACTIVE)
```

What to do next

You can define topics on one broker or queue manager that are available to publishers and subscribers on the connected queue managers. For more information, see Defining an administrative topic

Related concepts:

"Streams and topics" on page 35
Queued publish/subscribe has the concept of a publication stream that does not exist in the integrated publish/subscribe model. In queued publish/subscribe, streams provide a way of separating the flow of information for different topics. In IBM WebSphere MQ Version 6.0, a stream is implemented as a queue, defined at each broker that supports the stream. Each queue has the same name (the name of the stream). From IBM WebSphere MQ Version 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

Related reference:

"Introduction to IBM WebSphere MQ publish/subscribe messaging" on page 7
Publish/subscribe messaging allows you to decouple the provider of information, from the consumers of that information. The sending application and receiving application do not need to know anything about each other for the information to be sent and received.

Related information:

DISPLAY PUBSUB

Disconnect a queue manager from a broker hierarchy:

Disconnect a child queue manager from a parent queue manager in a broker hierarchy.

About this task

Use the **ALTER QMGR** command to disconnect a queue manager from a broker hierarchy. You can disconnect a queue manager in any order at any time.

The corresponding request to update the parent is sent when the connection between the queue managers is running.

Procedure

```
ALTER QMGR PARENT('')
```

Example

```
C:\Documents and Settings\Admin>runmqsc QM2
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM2.
    1 : alter qmgr parent('')
AMQ8005: WebSphere MQ queue manager changed.
    2 : display pubsub type(child)
AMQ8147: WebSphere MQ object  not found.
display pubsub type(parent)
    3 : display pubsub type(parent)
AMQ8147: WebSphere MQ object  not found.
```

What to do next

You can delete any streams, queues and manually defined channels that are no longer needed.

Publish/subscribe hierarchy example: Scenario 1:

Set up a publish/subscribe hierarchy topology using point-to-point channels with queue manager name alias.

About this task

These scenarios set up a publish/subscribe hierarchy in different ways to establish the connection between queue managers. These scenarios all use a parent queue manager called QM1, and two child queue managers called QM2, and QM3.

Scenario 1 is split into smaller sections to make the process easier to follow.

Scenario 1 part 1: Create the queue managers:

Procedure

1. Create and start three queue managers called QM1, QM2, and QM3 using the following commands:
   ```
   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
   strmqm QM1

   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
   strmqm QM2

   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
   strmqm QM3
   ```
2. Enable the queue manager publish/subscribe mode by using the following command on all three queue managers:
   ```
   ALTER QMGR PSMODE(ENABLED)
   ```

Scenario 1 part 2: Point-to-point channel connections:

About this task

Establish point-to-point channel connections between queue managers using a queue manager alias with the same name as the parent queue manager.

Procedure

1. Define a transmission queue and queue manager alias on QM2 to QM1. Define a sender channel to QM1 and a receiver channel for the sender channel created on QM1 for QM2:
   ```
   DEFINE QLOCAL(QM1.XMITQ) USAGE(XMITQ)

   DEFINE QREMOTE (QM1) RNAME('') RQMNAME(QM1) XMITQ(QM1.XMITQ)

   DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1.XMITQ) TRPTYPE(TCP)

   DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(RCVR) TRPTYPE(TCP)
   ```
2. Define a transmission queue and queue manager alias on QM3 to QM1. Define sender channel to QM1 and a receiver channel for the sender channel created on QM1 for QM3:
   ```
   DEFINE QLOCAL(QM1.XMITQ) USAGE(XMITQ)

   DEFINE QREMOTE (QM1) RNAME('') RQMNAME(QM1) XMITQ(QM1.XMITQ)

   DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1.XMITQ) TRPTYPE(TCP)

   DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(RCVR) TRPTYPE(TCP)
   ```
3. Define a transmission queue and queue manager alias on QM1 to QM2 and QM3. Define sender channel to QM2 and QM3, and a receiver channel for the sender channels created on QM2 and QM3 for QM1:
   ```
   DEFINE QLOCAL(QM2.XMITQ) USAGE(XMITQ)

   DEFINE QREMOTE (QM2) RNAME('') RQMNAME(QM2) XMITQ(QM2.XMITQ)
   ```

```
DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(SDR) CONNAME('localhost(7777)') XMITQ(QM2.XMITQ) TRPTYPE(TCP)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)

DEFINE QLOCAL(QM3.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM3) RNAME('') RQMNAME(QM3) XMITQ(QM3.XMITQ)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(SDR) CONNAME('localhost(8888)') XMITQ(QM3.XMITQ) TRPTYPE(TCP)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
```

4. Start the appropriate listeners on the queue managers:

   ```
   runmqlsr -m QM1 -t TCP -p 9999 &
   runmqlsr -m QM2 -t TCP -p 7777 &
   runmqlsr -m QM3 -t TCP -p 8888 &
   ```

5. Start the following channels:

   a. On QM1:

      ```
      START CHANNEL('QM1.TO.QM2')

      START CHANNEL('QM1.TO.QM3')
      ```

   b. On QM2:

      ```
      START CHANNEL('QM2.TO.QM1')
      ```

   c. On QM3:

      ```
      START CHANNEL('QM3.TO.QM1')
      ```

6. Check that all the channels have started:

   ```
   DISPLAY CHSTATUS('QM1.TO.QM2')

   DISPLAY CHSTATUS('QM1.TO.QM3')

   DISPLAY CHSTATUS('QM2.TO.QM1')

   DISPLAY CHSTATUS('QM3.TO.QM1')
   ```

Scenario 1 part 3: Connect queue managers and define a topic:

About this task

Connect the child queue managers QM2 and QM3 to the parent queue manager QM1.

Procedure

1. On QM2 and QM3, set the parent queue manager to QM1:

   ```
   ALTER QMGR PARENT (QM1)
   ```

2. Run the following command on all queue managers to check that the child queue managers are connected to the parent queue manager:

   ```
   DISPLAY PUBSUB TYPE(ALL)
   ```

3. Define a topic object:

   ```
   define topic(FOOTBALL) TOPICSTR('Sport/Soccer')
   ```

Scenario 1 part 4: Publish and subscribe the topic:

About this task

Use the amqspub.exe and amqssub.exe applications to publish and subscribe the topic.

Procedure

1. Run this command in the first command window:

   ```
   amqspub Sport/Soccer QM2
   ```

2. Run this command in the second command window:

   ```
   amqssub Sport/Soccer QM1
   ```

3. Run this command in the third command window:

   ```
   amqssub Sport/Soccer QM3
   ```

Results

The amqssub.exe applications in the second and third command windows receive the messages published in the first command window.

Related tasks:

"Publish/subscribe hierarchy example: Scenario 2"
Set up a publish/subscribe hierarchy topology using point-to-point channels with the transmission queue name the same as the remote queue manager.

"Publish/subscribe hierarchy example: Scenario 3" on page 80
Add a queue manager to a hierarchy topology using a cluster channel.

Publish/subscribe hierarchy example: Scenario 2:

Set up a publish/subscribe hierarchy topology using point-to-point channels with the transmission queue name the same as the remote queue manager.

About this task

These scenarios set up a publish/subscribe hierarchy in different ways to establish the connection between queue managers. These scenarios all use a parent queue manager called QM1, and two child queue managers called QM2, and QM3.

Scenario 2 is split into smaller sections to make the process easier to follow. This scenario reuses Scenario 1 part 1, Scenario 1 part 3, and Scenario 1 part 4 from "Publish/subscribe hierarchy example: Scenario 1" on page 76.

Scenario 2 part 1: Create queue manager and set PSMODE:

Procedure

1. Create and start three queue managers called QM1, QM2, and QM3 using the following commands:

   ```
   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
   strmqm QM1

   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
   strmqm QM2

   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
   strmqm QM3
   ```

2. Enable the queue manager publish/subscribe mode by using the following command on all three queue managers:

   ```
   ALTER QMGR PSMODE(ENABLED)
   ```

Scenario 2 part 2: Point-to-point channel connections:

About this task

Establish point-to-point channel connections between a queue manager using a transmission queue with the same name as the parent queue manager.

Procedure

1. Define a transmission queue on QM2 to QM1. Define a sender channel to QM1 and a receiver channel for the sender channel for QM2 created on QM1:

```
DEFINE QLOCAL(QM1) USAGE(XMITQ)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1) TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(RCVR) TRPTYPE(TCP)
```

2. Define a transmission queue on QM3 to QM1. Define sender channel to QM1 and a receiver channel for the sender channel created on QM1 for QM3:

   ```
   DEFINE QLOCAL(QM1) USAGE(XMITQ)

   DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1) TRPTYPE(TCP)

   DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(RCVR) TRPTYPE(TCP)
   ```

3. Define transmission queues on QM1 to QM2 and QM3. Define sender channels to QM2 and QM3, and a receiver channel for the sender channels created on QM2 and QM3 for QM1:

   ```
   DEFINE QLOCAL(QM2) USAGE(XMITQ)

   DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(SDR) CONNAME('localhost(7777)') XMITQ(QM2) TRPTYPE(TCP)

   DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)

   DEFINE QLOCAL(QM3) USAGE(XMITQ)

   DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(SDR) CONNAME('localhost(8888)') XMITQ(QM3) TRPTYPE(TCP)

   DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
   ```

4. Start the appropriate listeners on the queue managers:

   ```
   runmqlsr -m QM1 -t TCP -p 9999 &
   runmqlsr -m QM2 -t TCP -p 7777 &
   runmqlsr -m QM3 -t TCP -p 8888 &
   ```

5. Start the following channels:
   a. On QM1:

      ```
      START CHANNEL('QM1.TO.QM2')

      START CHANNEL('QM1.TO.QM3')
      ```

   b. On QM2:

      ```
      START CHANNEL('QM2.TO.QM1')
      ```

   c. On QM3:

      ```
      START CHANNEL('QM3.TO.QM1')
      ```

6. Check that all the channels have started:

   ```
   DISPLAY CHSTATUS('QM1.TO.QM2')

   DISPLAY CHSTATUS('QM1.TO.QM3')

   DISPLAY CHSTATUS('QM2.TO.QM1')

   DISPLAY CHSTATUS('QM3.TO.QM1')
   ```

Scenario 2 part 3: Connect queue managers and define a topic:

About this task

Connect the child queue managers QM2 and QM3 to the parent queue manager QM1.

Procedure

1. On QM2 and QM3, set the parent queue manager to QM1:

   ```
   ALTER QMGR PARENT (QM1)
   ```

2. Run the following command on all queue managers to check that the child queue managers are connected to the parent queue manager:

```
DISPLAY PUBSUB TYPE(ALL)
```

3. Define a topic object:

```
define topic(FOOTBALL) TOPICSTR('Sport/Soccer')
```

Scenario 2 part 4: Publish and subscribe the topic:

About this task

Use the amqspub.exe and amqssub.exe applications to publish and subscribe the topic.

Procedure

1. Run this command in the first command window:

```
amqspub Sport/Soccer QM2
```

2. Run this command in the second command window:

```
amqssub Sport/Soccer QM1
```

3. Run this command in the third command window:

```
amqssub Sport/Soccer QM3
```

Results

The amqssub.exe applications in the second and third command windows receive the messages published in the first command window.

Related tasks:

"Publish/subscribe hierarchy example: Scenario 1" on page 76
Set up a publish/subscribe hierarchy topology using point-to-point channels with queue manager name alias.

"Publish/subscribe hierarchy example: Scenario 3"
Add a queue manager to a hierarchy topology using a cluster channel.

Publish/subscribe hierarchy example: Scenario 3:

Add a queue manager to a hierarchy topology using a cluster channel.

About this task

These scenarios set up a publish/subscribe hierarchy in different ways to establish the connection between queue managers. These scenarios all use a parent queue manager called QM1, and two child queue managers called QM2, and QM3.

Scenario 3 is split into smaller sections to make the process easier to follow. This scenario reuses Scenario 1 part 1, Scenario 1 part 3, and Scenario 1 part 4 from "Publish/subscribe hierarchy example: Scenario 1" on page 76.

This scenario creates a cluster called DEMO where QM1 and QM2 are full repositories, and QM3 is a partial repository. Queue manager QM1 is the parent of queue managers QM2 and QM3.

Scenario 2 part 1: Create queue manager and set PSMODE:

Procedure

1. Create and start three queue managers called QM1, QM2, and QM3 using the following commands:

   ```
   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
   strmqm QM1

   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
   strmqm QM2

   crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
   strmqm QM3
   ```

2. Enable the queue manager publish/subscribe mode by using the following command on all three queue managers:

   ```
   ALTER QMGR PSMODE(ENABLED)
   ```

Scenario 2 part 2: Point-to-point channel connections:

About this task

Establish point-to-point channel connections between queue managers a cluster.

Procedure

1. On QM1 and QM2, set the **REPOS** parameter to the name of the cluster DEMO:

   ```
   ALTER QMGR REPOS(DEMO)
   ```

2. Start the appropriate listeners on the queue managers:

   ```
   runmqlsr -m QM1 -t TCP -p 9999 &
   runmqlsr -m QM2 -t TCP -p 7777 &
   runmqlsr -m QM3 -t TCP -p 8888 &
   ```

3. Define the cluster receiver channel on each queue manager:
   a. On QM1:

      ```
      DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(9999)') CLUSTER(DEMO)
      ```

   b. On QM2:

      ```
      DEFINE CHANNEL(TO.QM2) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(7777)') CLUSTER(DEMO)
      ```

   c. On QM3:

      ```
      DEFINE CHANNEL(TO.QM3) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(8888)') CLUSTER(DEMO)
      ```

4. Define a cluster sender channel to a full repository on each queue manager in the cluster:
   a. On QM1:

      ```
      DEFINE CHANNEL(TO.QM2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(7777)') CLUSTER(DEMO)
      ```

   b. On QM2:

      ```
      DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(9999)') CLUSTER(DEMO)
      ```

   c. QM3 can have a cluster sender channel to either full repository on QM1 or QM2. This example defines the channel to QM1:

      ```
      DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(9999)') CLUSTER(DEMO)
      ```

Scenario 2 part 3: Connect queue managers and define a topic:

About this task

Connect the child queue managers QM2 and QM3 to the parent queue manager QM1.

Procedure

1. On QM2 and QM3, set the parent queue manager to QM1:

   ```
   ALTER QMGR PARENT (QM1)
   ```

2. Run the following command on all queue managers to check that the child queue managers are connected to the parent queue manager:
   ```
   DISPLAY PUBSUB TYPE(ALL)
   ```
3. Define a topic object:
   ```
   define topic(FOOTBALL) TOPICSTR('Sport/Soccer')
   ```

Scenario 2 part 4: Publish and subscribe the topic:

About this task

Use the amqspub.exe and amqssub.exe applications to publish and subscribe the topic.

Procedure

1. Run this command in the first command window:
   ```
   amqspub Sport/Soccer QM2
   ```
2. Run this command in the second command window:
   ```
   amqssub Sport/Soccer QM1
   ```
3. Run this command in the third command window:
   ```
   amqssub Sport/Soccer QM3
   ```

Results

The amqssub.exe applications in the second and third command windows receive the messages published in the first command window.

Related tasks:

"Publish/subscribe hierarchy example: Scenario 1" on page 76
Set up a publish/subscribe hierarchy topology using point-to-point channels with queue manager name alias.

"Publish/subscribe hierarchy example: Scenario 2" on page 78
Set up a publish/subscribe hierarchy topology using point-to-point channels with the transmission queue name the same as the remote queue manager.

Controlling the flow of publications and subscriptions:

Queue managers that are connected together into a distributed publish/subscribe topology share a common federated topic space. You can control the flow of publications and subscriptions within the topology by choosing whether each publication and subscription is either local or global.

Local publications and subscriptions are not propagated beyond the queue manager to which the publisher or subscriber is connected.

You can control the extent of topic spaces created by connecting queue managers together in clusters or hierarchies. In a publish/subscribe cluster, the topic object must be 'clustered', or all elements stay local and the publication or subscription has no effect.

A subscription, when it matches topic strings in different publications, can resolve to different topic objects. These are called overlapping topics. The topic object that is associated with a publication for a particular match provides the topic attributes, and determines for example, if the subscriber is to receive the publication.

Publication scope:

The scope of a publication controls whether queue managers forward a publication to remote queue managers. Use the **PUBSCOPE** topic attribute to administer the scope of publications.

If a publication is not forwarded to remote queue managers, only local subscribers receive the publication.

The **PUBSCOPE** topic attribute is used to determine the scope of publications made to a specific topic. You can set the attribute to one of the following values:

QMGR
The publication is delivered only to local subscribers. These publications are called *local publications*. Local publications are not forwarded to remote queue managers and therefore are not received by subscribers connected to remote queue managers.

ALL
The publication is delivered to local subscribers and subscribers connected to remote queue managers. These publications are called *global publications*.

ASPARENT
Use the **PUBSCOPE** setting of the parent.

Publishers can also specify whether a publication is local or global using the MQPMO_SCOPE_QMGR put message option. If this option is used, it overrides any behavior that has been set using the **PUBSCOPE** topic attribute.

Subscription scope:

The scope of a subscription controls whether a subscription on one queue manager receives publications that are published on another queue manager in a publish/subscribe cluster or hierarchy, or only publications from local publishers.

Limiting the subscription scope to a queue manager stops proxy subscriptions from being forwarded to other queue managers in the publish/subscribe topology. This reduces inter-queue manager publish/subscribe messaging traffic.

The **SUBSCOPE** topic attribute is used to determine the scope of subscriptions that are made to a specific topic. You can set the attribute to one of the following values:

QMGR
A subscription receives only local publications, and proxy subscriptions are not propagated to remote queue managers.

ALL
A proxy subscription is propagated to remote queue managers, and the subscriber receives local and remote publications.

ASPARENT
Use the **SUBSCOPE** setting of the parent.

Individual subscribers can override the **SUBSCOPE** setting of ALL by specifying the MQSO_SCOPE_QMGR subscription option when creating a subscription. A subscription can override a topic's **SUBSCOPE** setting of ALL.

Note: Individual subscribers can only restrict the **SUBSCOPE** of the topic. When an individual subscription has **SUBSCOPE** set to ALL, the subscription honors the matching topics' **SUBSCOPE** setting.

Combining publication and subscription scopes:

In WebSphere MQ versions 7 onwards, publication and subscription scope work independently to determine the flow of publications between queue managers.

Publications can flow to all queue managers that are connected in a publish/subscribe topology, or only to the local queue manager. Similarly for proxy subscriptions. Which publications match a subscription is governed by the combination of these two flows.

Publications and subscriptions can both be scoped to `QMGR` or `ALL`. If a publisher and a subscriber are both connected to the same queue manager, scope settings do not affect which publications the subscriber receives from that publisher.

If the publisher and subscriber are connected to different queue managers, both settings must be `ALL` to receive remote publications.

Suppose publishers are connected to different queue managers. If you want a subscriber to receive publications from any publisher, set the subscription scope to `ALL`. You can then decide, for each publisher, whether to limit the scope of its publications to subscribers local to the publisher.

Suppose subscribers are connected to different queue managers. If you want the publications from a publisher to be sent to all the subscribers, set the publication scope to `ALL`. If you want a subscriber to receive publications only from a publisher connected to the same queue manager, set the subscription scope to `QMGR`.

In version 6, and earlier, publication and subscription scope not only governed which publications flowed. In addition the scope of the publication had to match the scope of the subscription. For the difference in the interpretation of scope between the version 6 and version 7 publish/subscribe, see Publish/Subscribe: Local and global publications and subscriptions.

Example: football results service

Suppose you are a member team in a football league. Each team has a queue manager connected to all the other teams in a publish/subscribe cluster.

The teams publish the results of all the games played on their home ground using the topic, `Football/result/`*`Home team name`*`/`*`Away team name`*. The strings in italics are variable topic names, and the publication is the result of the match.

Each club also republishes the results just for the club using the topic string `Football/myteam/`*`Home team name`*`/`*`Away team name`*.

Both topics are published to the whole cluster.

The following subscriptions have been set up by the league so that fans of any team can subscribe to the results in three interesting ways.

Notice that you can set up cluster topics with `SUBSCOPE(QMGR)`. The topic definitions are propagated to each member of the cluster, but the scope of the subscription is just the local queue manager. Thus subscribers at each queue manager receive different publications from the same subscription.

Receive all results

```
DEFINE TOPIC(A) TOPICSTR('Football/result/') CLUSTER SUBSCOPE(ALL)
```

Receive all home results

```
DEFINE TOPIC(B) TOPICSTR('Football/result/') CLUSTER SUBSCOPE(QMGR)
```

Because the subscription has QMGR scope, only results published at the home ground are matched.

Receive all my teams results

```
DEFINE TOPIC(C) TOPICSTR('Football/myteam/') CLUSTER SUBSCOPE(QMGR)
```

Because the subscription has QMGR scope, only the local team results, which are republished locally, are matched.

Topic spaces:

A topic space is the set of topics you can subscribe on. A subscriber connected to a queue manager in a distributed publish/subscribe topology has a topic space that potentially includes topics defined on connected queue managers.

Topics are initially created administratively, when you define a topic object or durable subscription, or dynamically when an application creates a publication or subscription dynamically.

Topics are propagated to other queue managers both through proxy subscriptions, and by creating administrative cluster topic objects. Proxy subscriptions result in publications being forwarded from the queue manager to which a publisher is connected, to the queue managers of subscribers. Proxy subscriptions are the mechanism by which topics defined on different queue managers are combined into a common topic space.

Proxy subscriptions are propagated between all queue managers that are connected together by parent-child relationships in a queue manager hierarchy. The result is, you can subscribe on one queue manager to a topic defined on any other queue manager in the hierarchy. As long as there is a connected path between the queue managers, it does not matter how the queue managers are connected.

Proxy subscriptions are also propagated for *cluster* topics between all the members of a cluster. A cluster topic is a topic that is attached to a topic object that has the **CLUSTER** attribute, or inherits the attribute from its parent. Topics that are not cluster topics are known as local topics and are not replicated to the cluster. No proxy subscriptions are propagated to the cluster from subscriptions to local topics.

To summarize, proxy subscriptions are created for subscribers in two circumstances.

1. A queue manager is a member of a hierarchy, and a proxy subscription is forwarded to the parent and children of the queue manager.
2. A queue manager is a member of a cluster, and the subscription topic string resolves to a topic that is associated with a cluster topic object. Proxy subscriptions are forwarded to all members of the cluster. See “Overlapping topics” on page 94 for more information about complications.

If a queue manager is a member of a cluster and a hierarchy, proxy subscriptions are propagated by both mechanisms without delivering duplicate publications to the subscriber.

The effect of creating a cluster topic object is twofold. Proxy subscriptions to a topic are sent to other members of the cluster when a subscription resolves to a cluster topic. It also sends a copy of the topic object to the other members of the cluster. The effect of forwarding cluster topic objects is to simplify the administration of topics. Typically, cluster topic objects are defined on a single queue manager in the cluster, called the cluster topic host.

The topics spaces of three publish/subscribe topologies are described in the following list:

- “Case 1. Publish/subscribe clusters” on page 86.
- “Case 2. Publish/subscribe hierarchies in version 7” on page 87.
- “Case 3. Publish/subscribe hierarchies and streams in version 6” on page 87.

In separate topics, the following tasks describe how to combine topic spaces.

- “Create a single topic space in a publish/subscribe cluster” on page 88.

- “Add a version 7 queue manager to existing version 6 topic spaces” on page 89.
- “Combine the topic spaces of multiple clusters” on page 90.
- “Combine and isolate topic spaces in multiple clusters” on page 91
- “Publish and subscribe to topic spaces in multiple clusters” on page 93

Case 1. Publish/subscribe clusters

In the example, assume that the queue manager is *not* connected to a publish/subscribe hierarchy.

If a queue manager is a member of a publish/subscribe cluster, its topic space is made up from local topics and cluster topics. Local topics are associated with topic objects without the **CLUSTER** attribute. If a queue manager has local topic object definitions, its topic space is different from another queue manager in the cluster that also has its own locally defined topic objects.

In a publish/subscribe cluster, you cannot subscribe to a topic defined on another queue manager, unless the topic you subscribe to resolves to a cluster topic object.

Conflicting definitions of a cluster topic defined elsewhere in the cluster are resolved in favor of the most recent definition. At any point in time, if a cluster topic has been multiply defined, the cluster topic definition on different queue managers might be different.

A local definition of a topic object, whether the definition is for a cluster topic or a local topic, takes precedence over the same topic object defined elsewhere in the cluster. The locally defined topic is used, even if the object defined elsewhere is more recent.

Set either of the **PUBSCOPE** and **SUBSCOPE** options to QMGR, to prevent a publication or a subscription on a cluster topic flowing to different queue managers in the cluster.

Suppose you define a cluster topic object Alabama with the topic string USA/Alabama on your cluster topic host. The result is as follows:

1. The topic space at the cluster topic host now includes the cluster topic object Alabama and the topic USA/Alabama.
2. The cluster topic object Alabama is replicated to all queue managers in the cluster where it is combined with the topic space at each queue manager. What happens at each queue manager in the cluster depends on whether the topic object Alabama exists at a queue manager.
   - If Alabama is a new topic object, the queue manager adds the cluster topic object Alabama, and the topic USA/Alabama, to its topic space.
   - If Alabama is a local definition, the cluster topic object Alabama is added. Unless the local definition is deleted, the remotely defined cluster topic object is ignored. The queue manager retains both definitions.
   - If Alabama is an older cluster topic object defined elsewhere, it is replaced by the newer cluster topic object.
3. An application or administrator, anywhere in the cluster, can create a subscription to USA/Alabama by referring to the Alabama topic object.
4. An application, anywhere in the cluster, using the topic string USA/Alabama directly can create a subscription that inherits the attributes of the topic object Alabama. The Alabama topic object is inherited by a subscription formed from any topic string beginning with USA/Alabama,

   If there is another definition of the Alabama topic object on one of the other queue managers, it takes precedence over the definition on the cluster topic host. The local object might have a cluster attribute, or it might not. The cluster attribute might refer to the same cluster or another cluster. Try to avoid these multiple definition cases. They lead to differences in behavior.
5. If the topic object Alabama has the **PUBSCOPE** attribute ALL, subscriptions that resolve to Alabama are sent to all the other queue managers in the cluster.

Set the Alabama **PUBSCOPE** attribute to QMGR to prevent publications flowing from publishers to subscribers connected to different queue managers in the cluster.

The Alabama topic object is replicated to every queue manager in the cluster, so the **PUBSCOPE** and **PUBSCOPE** attributes apply to all the queue managers in the cluster.

It is important that a cluster topic object is associated with the same topic string everywhere in the cluster. You cannot modify the topic string with which a topic object is associated. To associate the same topic object with a different topic string, you must delete the topic object and re-create it with the new topic string. If the topic is clustered, the effect is to delete the copies of the topic object stored on the other members of the cluster and then to create copies of the new topic object everywhere in the cluster. The copies of the topic object all refer to the same topic string.

However, you can create a duplicate definition of a topic object on another queue manager in the cluster, with a different topic string. Always try to avoid duplicates by managing cluster topic hosts on one queue manager. See "Multiple cluster topic definitions" on page 63 for more information on this important point. Multiple definitions of the same topic object with different topic strings can produce different results depending how and where the topic is referenced.

Case 2. Publish/subscribe hierarchies in version 7

In the example, assume that the queue manager is *not* a member of a publish/subscribe cluster.

In version 7, if a queue manager is a member of a publish/subscribe hierarchy, its topic space consists of all the topics defined locally and on connected queue managers. The topic space of all the queue managers in a hierarchy is the same. There is no division of topics into local topics and cluster topics.

Set either of the **PUBSCOPE** and **SUBSCOPE** options to QMGR, to prevent a publication on a topic flowing from a publisher to a subscriber connected to different queue managers in the hierarchy.

Suppose you define a topic object Alabama with the topic string USA/Alabama on queue manager QMA. The result is as follows:

1. The topic space at QMA now includes the topic object Alabama and the topic string USA/Alabama.
2. An application or administrator can create a subscription at QMA using the topic object name Alabama.
3. An application can create a subscription to any topic, including USA/Alabama, at any queue manager in the hierarchy. If QMA has not been defined locally, the topic USA/Alabama resolves to the topic object SYSTEM.BASE.TOPIC.

Case 3. Publish/subscribe hierarchies and streams in version 6

Before version 7, the topic space was divided into separate streams, which included the default stream that was present on all queue managers. Publications cannot flow between different streams. If named streams are used, the topic spaces at different queue managers might be different. Topics are divided into topics in the default stream, and topics in different named streams.

Note: Each named stream forms a separate topic space. To form a connected topology each named stream must exist on the connected queue managers. Suppose stream X is defined on QMA and QMC, but not on QMB. If QMA is the parent of QMB, and QMB is the parent of QMC, no topics in stream X can flow between QMA and QMC.

Setting both of the **PUBSCOPE** and **SUBSCOPE** options either to QMGR or to ALL requires a publisher and subscriber to a topic to exchange only publications for local consumption, or to exchange only publications for global consumption.

From version 7, streams are not available using the publish/subscribe API. If you use queued publish/subscribe on a version 7 queue manager, streams are mapped to different topic objects that can simulate the effect of streams. A stream is simulated by creating a topic object that is the root topic for all the topics in the stream. The queue manager maps publications and subscriptions between the stream and the corresponding root topic of each tree.

Combining topics spaces:

Combine the topic space of a queue manager with other queue managers in a publish/subscribe cluster or hierarchy. Combine publish/subscribe clusters, and publish/subscribe clusters with hierarchies.

You can create different publish/subscribe topics spaces by using the building blocks of **CLUSTER**, **PUBSCOPE** and **SUBSCOPE** attributes, publish/subscribe clusters, and publish/subscribe hierarchies.

Starting from the example of scaling up from a single queue manager to a publish/subscribe cluster, the following scenarios illustrate different publish/subscribe topologies.

Create a single topic space in a publish/subscribe cluster:

Scale up a publish/subscribe system to run on multiple queue managers. Use a publish/subscribe cluster to provide each publisher and subscriber with a single identical topic space.

Before you begin

You have implemented a publish/subscribe system on a single version 7 queue manager.

Always create topic spaces with their own root topics, rather than relying on inheriting the attributes of SYSTEM.BASE.TOPIC. If you scale your publish/subscribe system up to a cluster, you can define your root topics as cluster topics, on the cluster topic host, and then all your topics are shared throughout the cluster.

About this task

You now want to scale the system up to support more publishers and subscribers and have every topic visible throughout the cluster.

Procedure

1. Create a cluster to use with the publish/subscribe system. If you have an existing traditional cluster, for performance reasons it is better to set up a new cluster for the new publish subscribe system. You can use the same servers for the cluster repositories of both clusters
2. Choose one queue manager, possibly one of the repositories, to be the cluster topic host.
3. Ensure every topic that is to be visible throughout the publish/subscribe cluster resolves to an administrative topic object. Set the **CLUSTER** attribute naming the publish/subscribe cluster.

What to do next

Connect publisher and subscriber applications to any queue managers in the cluster.

Create administrative topic objects that have the **CLUSTER** attribute. The topics are also propagated throughout the cluster. Publisher and subscriber programs use the administrative topics so that their behavior is not altered by being connected to different queue managers in the cluster

If you need SYSTEM.BASE.TOPIC to act like a cluster topic on every queue manager, you need to modify it on every queue manager.

Add a version 7 queue manager to existing version 6 topic spaces:

Extend an existing version 6 publish/subscribe system to interoperate with a version 7 queue manager, sharing the same topic spaces.

Before you begin

You have an existing version 6 publish/subscribe system.

You have installed WebSphere MQ version 7 on a new server and configured a queue manager.

About this task

You want to extend your existing version 6 publish/subscribe system to work with version 7 queue managers.

You have decided to stabilize development of the version 6 publish/subscribe system that uses the queued publish/subscribe interface. You intend to add extensions to the system using the version 7 MQI. You have no plans now to rewrite the queued publish/subscribe applications.

You intend to upgrade the version 6 queue managers to version 7 in the future. For now, you are continuing to run the existing queued publish/subscribe applications on the version 7 queue managers.

Procedure

1. Create one set of sender-receiver channels to connect the version 7 queue manager with one of the version 6 queue managers in both directions.
2. Create two transmission queues with the names of the target queue managers. Use queue manager aliases if you cannot use the name of the target queue manager as the transmission queue name for some reason.
3. Configure the transmission queues to trigger the sender channels.
4. If the version 6 publish/subscribe system uses streams, add the streams to the version 7 queue manager as described in Adding a stream.
5. Check the version 7 queue manager **PSMODE** is set to ENABLE.
6. Alter its **PARENT** attribute to refer to one of the version 6 queue managers.
7. Check the status of the parent-child relationship between the queue managers is active in both directions.

What to do next

Once you have completed the task, both the version 6 and version 7 queue manager share the same topic spaces. For example, you can do all the following tasks.

- Exchange publications and subscriptions between version 6 and version 7 queue managers.
- Run your existing version 6 publish/subscribe programs on the version 7 queue manager.
- View and modify the topic space on either the version 6 or version 7 queue manager.
- Write version 7 publish/subscribe applications and run them on the version 7 queue manager.
- Create new publications and subscriptions with the version 7 applications and exchange them with version 6 applications.

Combine the topic spaces of multiple clusters:

Create topic spaces that span multiple clusters. Publish to a topic in one cluster and subscribe to it in another.

Before you begin

You have existing publish/subscribe clusters, and you want to propagate some cluster topics into all the clusters.

About this task

To propagate publications from one cluster to another, you need to join the clusters together in a hierarchy; see Figure 35. The hierarchical connections propagate subscriptions and publications between the connected queue managers, and the clusters propagate cluster topics within each cluster, but not between clusters.

The combination of these two mechanisms propagates cluster topics between all the clusters. You need to repeat the cluster topic definitions in each cluster.

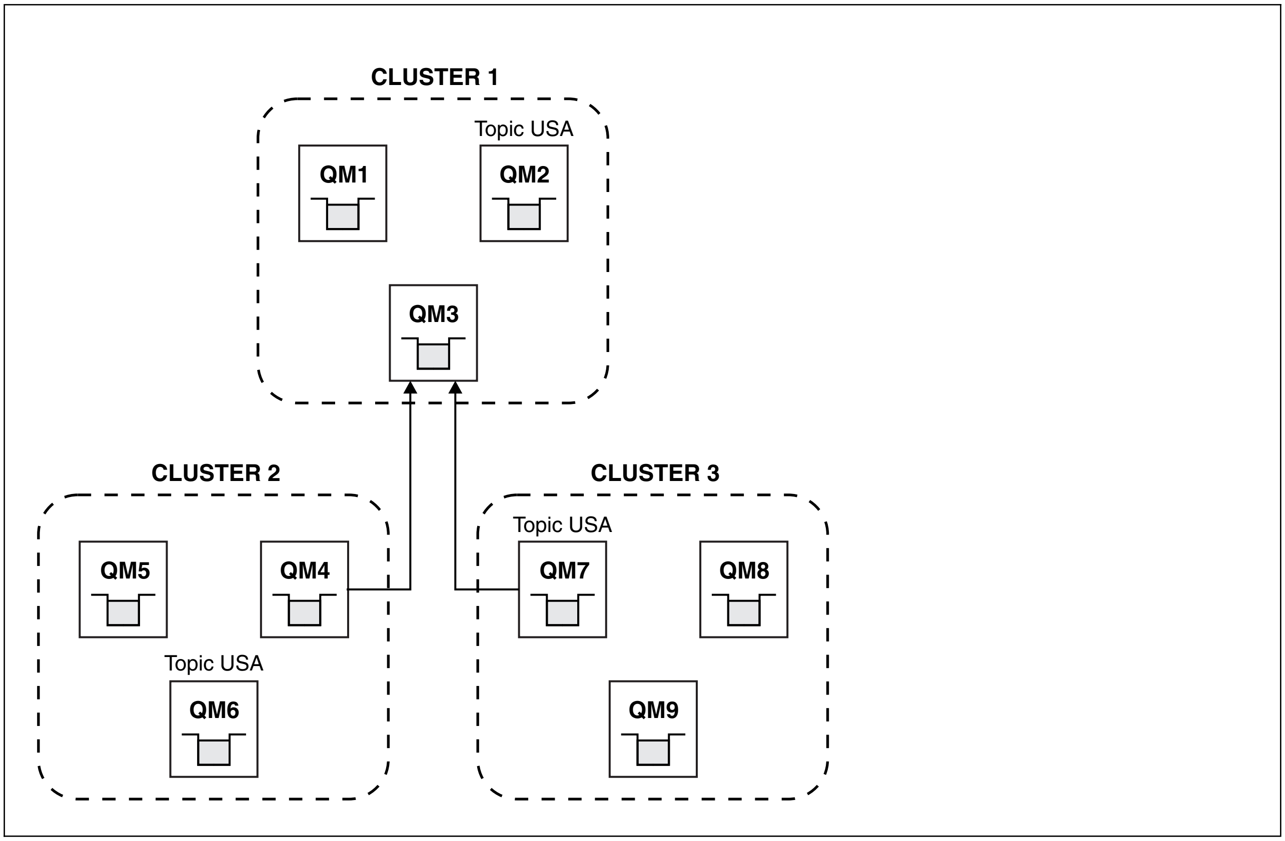


Figure 35. Connecting clusters using hierarchies

The following steps connect the clusters into a hierarchy.

Procedure

1. Create two sets of sender-receiver channels to connect QM3 and QM4, and QM3 and QM7, in both directions. You must use traditional sender-receiver channels and transmission queues, rather than a cluster, to connect a hierarchy.

2. Create three transmission queues with the names of the target queue managers. Use queue manager aliases if you cannot use the name of the target queue manager as the transmission queue name for some reason.
3. Configure the transmission queues to trigger the sender channels.
4. Check the **PSMODE** of QM3, QM4 and QM7 is set to ENABLE.
5. Alter the **PARENT** attribute of QM4 and QM7 to QM3.
6. Check the status of the parent-child relationship between the queue managers is active in both directions.
7. Create the administrative topic USA with the attribute **CLUSTER**('CLUSTER 1'), **CLUSTER**('CLUSTER 2'), and **CLUSTER**('CLUSTER 3') on each of the three cluster topic hosts in clusters 1, 2 and 3. The cluster topic host does not need to be a hierarchically connected queue manager.

What to do next

You can now publish or subscribe to the cluster topic USA in Figure 35 on page 90. The publications subscriptions flow to publishers and subscribers in all three clusters.

Suppose that you did not create USA as a cluster topic in the other clusters. If USA is only defined on QM7, then publications and subscriptions to USA are exchanged between QM7, QM8, QM9, and QM3. Publishers and subscribers running on QM7, QM8, QM9 inherit the attributes of the administrative topic USA. Publishers and subscribers on QM3 inherit the attributes of SYSTEM.BASE.TOPIC on QM3.

Combine and isolate topic spaces in multiple clusters:

Isolate some topic spaces to a specific cluster, and combine other topic spaces to make them accessible in all the connected clusters.

Before you begin

Examine the topic "Combine the topic spaces of multiple clusters" on page 90. It might be sufficient for your needs, without adding an additional queue manager as a bridge.

About this task

A potential improvement on the topology shown in Figure 35 on page 90 in "Combine the topic spaces of multiple clusters" on page 90 is to isolate cluster topics that are not shared across all the clusters. Isolate clusters by creating a bridging queue manager that is not in any of the clusters; see Figure 36 on page 92. Use the bridging queue manager to filter which publications and subscriptions can flow from one cluster to another.

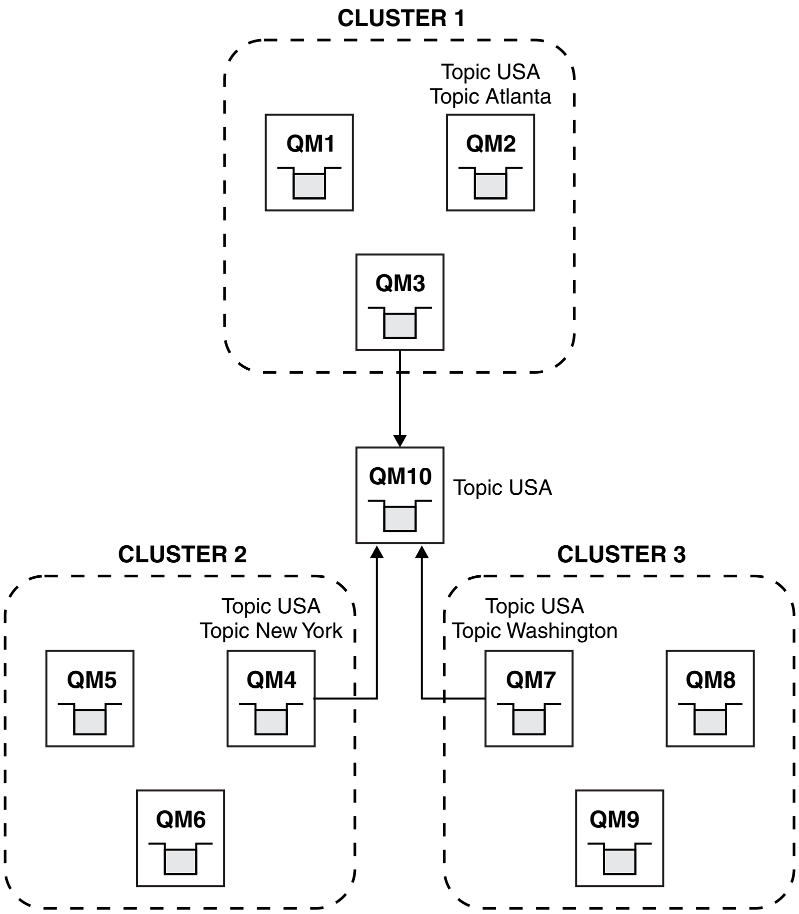


Figure 36. Bridged clusters

Use the bridge to isolate cluster topics that you do not want exposed across the bridge on the other clusters. In Figure 36, `USA` is a cluster topic shared in all the clusters, and `Atlanta`, `New York` and `Washington` are cluster topics that are shared only in one cluster each.

Model your configuration using the following procedure:

Procedure

1. Modify all the `SYSTEM.BASE.TOPIC` topic objects to have **SUBSCOPE**(QMGR) and **PUBSCOPE**(QMGR) on all the queue managers. No topics (even cluster topics) are propagated onto other queue managers unless you explicitly set **SUBSCOPE**(ALL) and **PUBSCOPE**(ALL) on the root topic of your cluster topics.
2. Define the topics on the three cluster topic hosts that you want to be shared in each cluster with the attributes **CLUSTER**(*clustername*), **SUBSCOPE**(ALL) and **PUBSCOPE**(ALL). If you want some cluster topics shared between all the clusters, define the same topic in each of the clusters. Use the cluster name of each cluster as the cluster attribute.
3. For the cluster topics you want shared between all the clusters, define the topics again on the bridge queue manager (QM10), with the attributes **SUBSCOPE**(ALL), and **PUBSCOPE**(ALL).

Example

In the example in Figure 36, only topics that inherit from `USA` propagate between all three clusters.

What to do next

Subscriptions for topics defined on the bridge queue manager with **SUBSCOPE**(ALL) and **PUBSCOPE**(ALL) are propagated between the clusters.

Subscriptions for topics defined within each cluster with attributes **CLUSTER**(*clustername*), **SUBSCOPE**(ALL) and **PUBSCOPE**(ALL) are propagated within each cluster.

Any other subscriptions are local to a queue manager.

Publish and subscribe to topic spaces in multiple clusters:

Publish and subscribe to topics in multiple clusters using overlapped clusters. You can use this technique as long as the topic spaces in the clusters do not overlap.

Before you begin

Create multiple traditional clusters with some queue managers in the intersections between the clusters.

About this task

You might have chosen to overlap clusters for various different reasons.

1. You have a limited number of high availability servers, or queue managers. You decide to deploy all the cluster repositories, and cluster topic hosts to them.
2. You have existing traditional queue manager clusters that are connected using gateway queue managers. You want to deploy publish/subscribe applications to the same cluster topology.
3. You have a several self contained publish/subscribe applications. For performance reasons, it is better to keep publish/subscribe clusters small and separate from traditional clusters. You have decided to deploy the applications to different clusters. However, you also want to monitor all the publish/subscribe applications on one queue manager, as you have licensed only one copy of the monitoring application. This queue manager must have access to the publications to cluster topics in all the clusters.

By ensuring that your topics are defined in non-overlapping topic spaces, you can deploy the topics into overlapping publish/subscribe clusters, see Figure 37 on page 94. If the topic spaces overlap, then deploying to overlapping clusters leads to problems.

Because the publish/subscribe clusters overlap you can publish and subscribe to any of the topic spaces using the queue managers in the overlap.

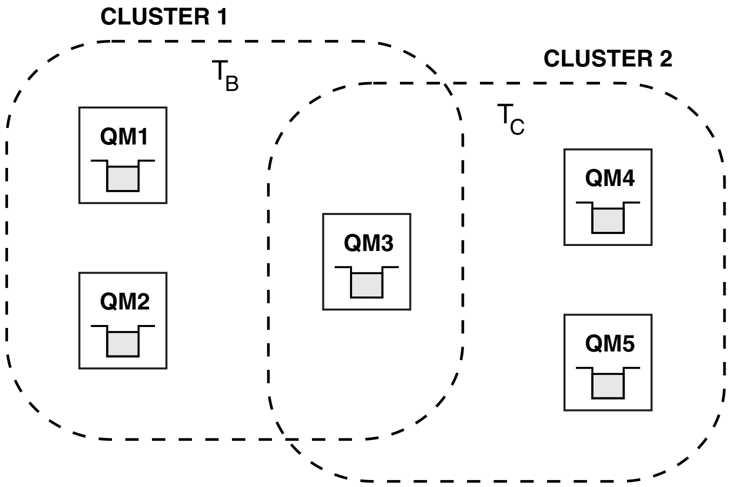


Figure 37. Overlapping clusters, non-overlapping topic spaces

Procedure

Create a means of ensuring that topic spaces do not overlap. For example, define a unique root topic for each of the topic spaces. Make the root topics cluster topics.

1. `DEFINE TOPIC(B) TOPICSTR('B') CLUSTER('CLUSTER 1') ...`
2. `DEFINE TOPIC(C) TOPICSTR('C') CLUSTER('CLUSTER 2') ...`

Example

In Figure 37 publishers and subscriber connected to `QM3` can publish or subscribe to T_B or T_C

What to do next

Connect publishers and subscribers that use topics in both clusters to queue managers in the overlap.

Connect publishers and subscribers that must only use topics in a specific cluster to queue managers not in the overlap.

Overlapping topics:

Overlapping topics occur when a publication can be associated with different topic objects, depending on the distributed publish/subscribe topology, the publication, and the subscription topic strings.

Overlaps between topics have to be considered if a topic could be resolved to more than one topic object.

Local topics in a cluster
A topic can be defined on any queue manager in a cluster. If the topic is defined locally, then it takes precedence over a cluster topic that is defined elsewhere and resolves to the same topic string.

Cluster topics in a cluster
A topic can be defined on any queue manager in a cluster. If the topic is clustered, then it is replicated to other members of the cluster. If the topic is defined as a cluster topic on another queue manager in the cluster, it is an error. An error message is written to the error log of the queue manager that has an existing cluster definition.

As a rule, define cluster topics only on one queue manager in the cluster, the "cluster topic host", to ensure there is only one definition of a cluster topic.

If you redefine a cluster topic, the change takes time to reach each queue manager. Eventually, the latest definition overrides the earlier cluster topic definitions that have been replicated to non-cluster topic hosts.

If you define a cluster topic on multiple queue managers in the cluster with different attributes, the latest definition does not override any earlier local definitions.

Wildcard subscriptions resolve to multiple topic strings

When a subscription contains wildcards, potentially different topics in a topic space can match the subscription and result in the subscription resolving to different topic objects.

For example, consider the following topic definitions in the cluster SPORTS.

```
DEFINE TOPIC(A) TOPICSTR('Football/result/#') SUBSCOPE(QMGR) CLUSTER(SPORTS)
DEFINE TOPIC(B) TOPICSTR('Football/#') SUBSCOPE(ALL) CLUSTER(SPORTS)
DEFINE TOPIC(C) TOPICSTR('Football/result/Newport/Cardiff') PUBSCOPE(ALL) SUBSCOPE(ALL) CLUSTER(SPORTS)
DEFINE TOPIC(D) TOPICSTR('Football/matches/Newport/Cardiff') PUBSCOPE(ALL) SUBSCOPE(QMGR) CLUSTER(SPORTS)
```

Suppose there are two queue managers QM1 and QM2 in the cluster. Topics C and D are published on QM1.

Consider what a subscriber on QM2 receives, if these subscriptions are ungrouped.

- A subscription to topic A receives nothing.
  - SUBSCOPE(QMGR), and the publication is on the other queue manager.
- A subscription to topic B receives both publications.
  - SUBSCOPE(ALL) and PUBSCOPE(ALL) in both cases.
- A subscription to topic C receives one publication.
  - SUBSCOPE(ALL) and PUBSCOPE(ALL), and a match to the publication on topic C.
- A subscription to topic D receives nothing.
  - SUBSCOPE(QMGR), and the publication is on the other queue manager.

Consider what a subscriber on QM2 receives, if these subscriptions are grouped.

- The subscriber receives one publication on topic C.
  - The matching subscription on topic A with SUBSCOPE(QMGR) is overridden by the matching subscription on topic C with SUBSCOPE(ALL). The more specific subscription wins, and the publication is received.
  - The matching subscription on topic B is rejected in favor of the matching subscription on topic C, because the subscriptions are grouped, and C is more specific. The duplicate publication is discarded.
- The subscriber receives no publication on topic D
  - The matching subscription on topic B with SUBSCOPE(ALL) is overridden by the matching subscription on topic D with SUBSCOPE(QMGR). The more specific subscription wins, and the publication is discarded.

How loop detection works:

In a distributed publish/subscribe network, it is important that publications and proxy subscriptions cannot loop, as this would result in a flooded network with connected subscribers receiving multiple copies of the same original publication.

The proxy subscription aggregation system described in “Proxy subscription aggregation and publication aggregation” on page 43 does not prevent the formation of a loop, although it will prevent the perpetual looping of proxy subscriptions. Because the propagation of publications is determined by the existence of proxy subscriptions, they can enter a perpetual loop. Websphere MQ V7.0 uses the following technique to prevent publications from perpetually looping:

As publications move around a publish/subscribe topology each queue manager adds a unique fingerprint to the message header. Whenever a publish/subscribe queue manager receives a publication from another publish/subscribe queue manager, the fingerprints held in the message header are checked. If its own fingerprint is already present, the publication has fully circulated around a loop, so the queue manager discards the message, and adds an entry to the error log.

Note: Within a loop, publications are propagated in both directions around the loop, and each queue manager within the loop receives both publications before the originating queue manager discards the looped publications. This results in subscribing applications receiving duplicate copies of publications until the loop is broken.

Loop detection fingerprint format:

The loop detection fingerprints are inserted into an RFH2 header or flow as part of the V7.0 protocol. An RFH2 programmer needs to understand the header and pass on the fingerprint information intact. WebSphere MessageBroker uses RFH1 headers which will not contain the fingerprint information.

```
<ibm>
  <Rfp>uuid1</Rfp>
  <Rfp>uuid2</Rfp>
  <Rfp>uuid3</Rfp>
  . . .
</ibm>
```

<ibm> is the name of the folder that holds the list of routing fingerprints containing the unique user identifier (uuid) of each queue manager that has been visited.

Every time that a message is published by a queue manager, it adds its uuid into the <ibm> folder using the <Rfp> (routing fingerprint) tag. Whenever a publication is received, WebSphere MQ uses the message properties API to iterate through the <Rfp> tags to see if that particular uuid value is present. Because of the way that the WebSphere Platform Messaging component of WebSphere MQ attaches to Websphere Message Broker through a channel and RFH2 subscription when using the queued publish/subscribe interface, WebSphere MQ also creates a fingerprint when it receives a publication by that route.

The goal is to not deliver any RFH2 to an application if it is not expecting any, simply because we have added in our fingerprint information.

Whenever an RFH2 is converted into message properties, it will also be necessary to convert the <ibm> folder; this removes the fingerprint information from the RFH2 that is passed on or delivered to applications that have used the Websphere MQ V7.0 API.

Whenever a message that has fingerprint information is delivered to an RFH1 subscriber or is passed onto Websphere Message Broker V6.0, the fingerprint information is converted to an RFH1.

When Websphere Message Broker V6.0 passes this message to an RFH2 subscriber, such as SIB, it has to convert the fingerprint information back to an RFH2 format.

JMS applications do not see the fingerprint information, because the JMS interface does not extract that information from the RFH2, and therefore does not hand it on to its applications.

The Rfp message properties are created with `propDesc.CopyOptions =  MQCOPY_FORWARD and MQCOPY_PUBLISH`. This has implications for applications receiving and then republishing the same message. It means that such an application can continue the chain of routing fingerprints by using `PutMsgOpts.Action = MQACTP_FORWARD`, but must be coded appropriately to remove its own fingerprint from the chain. By default the application uses `PutMsgOpts.Action = MQACTP_NEW` and starts a new chain.

Retained publications in a distributed publish/subscribe topology:

When using retained publications in a distributed publish/subscribe topology, it is best practice to publish only retained publications on the same topic from a single queue manager in the topology.

Otherwise, it is possible that different retained publications might be active at different queue managers for the same topic, leading to unexpected behavior. As multiple proxy subscriptions are distributed, multiple retained publications might be received.

Publish/subscribe security between queue managers:

Publish/subscribe internal messages, such as proxy subscriptions and publications, are put to publish/subscribe system queues using normal channel security rules. The information and diagrams in this topic highlight the various processes and user IDs involved in the delivery of these messages.

Local access control

Access to topics for publication and subscriptions is governed by local security definitions and rules that are described in Publish/subscribe security. On z/OS, no local topic object is required to establish access control. No local topic is required for access control on other platforms either. Administrators can choose to apply access control to clustered topic objects, irrespective of whether they exist in the cluster yet.

System administrators are responsible for access control on their local system. They must trust the administrators of other members of the hierarchy or cluster collectives to be responsible for their access control policy. Because access control is defined for each separate machine it is likely to be burdensome if fine level control is needed. It might not be necessary to impose any access control, or access control might be defined on high-level objects in the topic tree. Fine level access control can be defined for each subdivision of the topic namespace.

Making a proxy subscription

Trust for an organization to connect its queue manager to your queue manager is confirmed by normal channel authentication means. If that trusted organization is also allowed to do distributed publish/subscribe, an authority check is done. The check is made when the channel puts a message to a distributed publish/subscribe queue. For example, if a message is put to the `SYSTEM.INTER.QMGR.CONTROL` queue. The user ID for the queue authority check depends on the `PUTAUT` values of the receiving channel. For example, the user ID of the channel, `MCAUSER`, the message context, depending on the value and platform. For more information about channel security, see Channel security.

Proxy subscriptions are made with the user ID of the distributed publish/subscribe agent on the remote queue manager. For example, `QM2` in Figure 38 on page 98. The user is then easily granted access to local topic object profiles, because that user ID is defined in the system and there are therefore no domain conflicts.

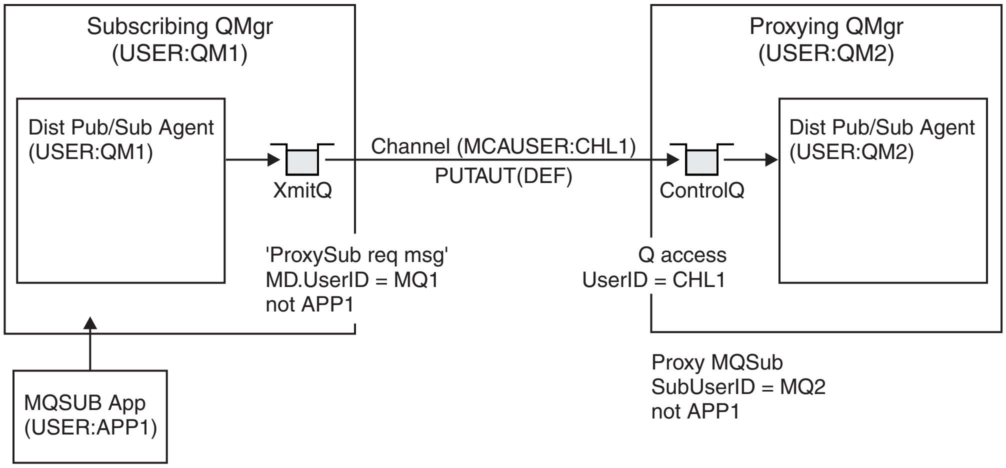


Figure 38. Proxy subscription security, making a subscription

Sending back remote publications

When a publication is created on the publishing queue manager, a copy of the publication is created for any proxy subscription. The context of the copied publication contains the context of the user ID which made the subscription; QM2 in Figure 39 on page 99. The proxy subscription is created with a destination queue that is a remote queue, so the publication message is resolved onto a transmission queue.

Trust for an organization to connect its queue manager, QM2, to another queue manager, QM1, is confirmed by normal channel authentication means. If that trusted organization is then allowed to do distributed publish/subscribe, an authority check is done when the channel puts the publication message to the distributed publish/subscribe publication queue SYSTEM.INTER.QMGR.PUBS. The user ID for the queue authority check depends on the PUTAUT value of the receiving channel (for example, the user ID of the channel, MCAUSER, message context, and others, depending on value and platform). For more information about channel security, see Channel security.

When the publication message reaches the subscribing queue manager, another MQPUT to the topic is done under the authority of that queue manager and the context with the message is replaced by the context of each of the local subscribers as they are each given the message.

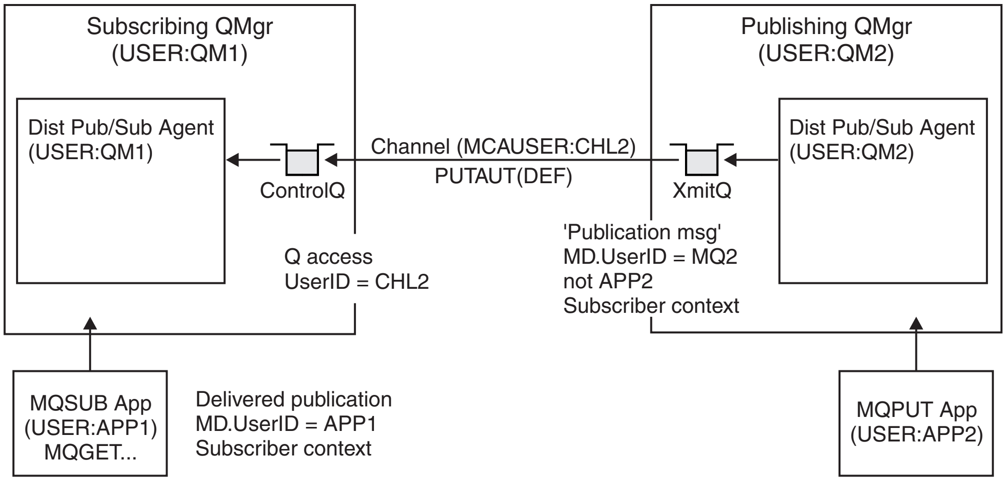


Figure 39. Proxy subscription security, forwarding publications

On a system where little has been considered regarding security, the distributed publish/subscribe processes are likely to be running under a user ID in the `mqm` group, the `MCAUSER` parameter on a channel is blank (the default), and messages are delivered to the various system queues as required. The unsecured system makes it easy to set up a proof of concept to demonstrate distributed publish/subscribe.

On a system where security is more seriously considered, these internal messages are subject to the same security controls as any message going over the channel.

If the channel is set up with a non-blank `MCAUSER` and a `PUTAUT` value specifying that `MCAUSER` must be checked, then the `MCAUSER` in question must be granted access to `SYSTEM.INTER.QMGR.*` queues. If there are multiple different remote queue managers, with channels running under different `MCAUSER` IDs, all those user IDs need to be granted access to the `SYSTEM.INTER.QMGR.*` queues. Channels running under different `MCAUSER` IDs might occur, for example, when multiple hierarchical connections are configured on a single queue manager.

If the channel is set up with a `PUTAUT` value specifying that the context of the message is used, then access to the `SYSTEM.INTER.QMGR.*` queues are checked based on the user ID inside the internal message. Because all these messages are put with the user ID of the distributed publish/subscribe agent from the queue manager that is sending the internal message, or publication message (see Figure 39), it is not too large a set of user IDs to grant access to the various system queues (one per remote queue manager), if you want to set up your distributed publish/subscribe security in this way. It still has all the same issues that channel context security always has; that of the different user ID domains and the fact that the user ID in the message might not be defined on the receiving system. However, it is a perfectly acceptable way to run if required.

All inter-queue manager messaging for the purposes of distributed publish/subscribe runs using normal channel security.

For information about restricting publications and proxy subscriptions at the topic level, see Publish/subscribe security.

Using default user IDs with a queue manager hierarchy

If you have a hierarchy of queue managers running on different platforms and are using default user IDs, note that these default user IDs differ between platforms and might not be known on the target platform.

As a result, a queue manager running on one platform rejects messages received from queue managers on other platforms with the reason code MQRC_NOT_AUTHORIZED.

To avoid messages being rejected, at a minimum, the following authorities need to be added to the default user IDs used on other platforms:

- *PUT *GET authority on the SYSTEM.BROKER. queues
- *PUB *SUB authority on the SYSTEM.BROKER. topics
- *ADMCRT *ADMDLT *ADMCHG authority on the SYSTEM.BROKER.CONTROL.QUEUE queue.

The default user IDs are as follows:

Platform	Default user ID
Windows	MUSR_MQADMIN **Note:** MUSR_MQADMIN is the default user ID for the first installation only. For subsequent installations, the Prepare IBM WebSphere MQ Wizard creates a user account named forMUSR_MQADMINx, where x is the next available number representing a user ID that does not exist.
UNIX and Linux systems	mqm
IBM i	QMQM
z/OS	The channel initiator address space user ID

Create and grant access to the 'qmqm' user ID if hierarchically attached to a queue manager on IBM i for Queue Managers on Windows, UNIX, Linux, and z/OS platforms.

Create and grant access to the 'mqm' user ID if hierarchically attached to a queue manager on Windows, UNIX, or Linux for Queue Managers on IBM i and z/OS platforms.

Create and grant user access to the z/OS channel initiator address space user ID if hierarchically attached to a queue manager on z/OS for Queue Managers on Windows, UNIX, Linux, and IBM i platforms.

User IDs can be case sensitive. The originating queue manager (if IBM i, Windows, UNIX, or Linux systems) forces the user ID to be all uppercase. The receiving queue manager (if Windows, UNIX or Linux systems) forces the user ID to be all lowercase. Therefore, all user IDs created on UNIX and Linux systems must be created in their lowercase form. If a message exit has been installed, forcing the user ID into uppercase or lowercase does not take place. Care must be taken to understand how the message exit processes the user ID.

To avoid potential problems with conversion of user IDs:

- On UNIX, Linux and Windows systems, ensure the user IDs are specified in lowercase.
- On IBM i and z/OS, ensure the user IDs are specified in uppercase.

Distributed publish/subscribe system queues:

Four system queues are used by queue managers for publish/subscribe messaging. You need to be aware of their existence only for problem determination or capacity planning purposes.

Table 9. Publish/subscribe system queues on distributed platforms

System queue	Purpose
SYSTEM.INTER.QMGR.CONTROL	WebSphere MQ distributed publish/subscribe control queue
SYSTEM.INTER.QMGR.FANREQ	WebSphere MQ distributed publish/subscribe internal proxy subscription fan-out process input queue
SYSTEM.INTER.QMGR.PUBS	WebSphere MQ distributed publish/subscribe publications
SYSTEM.HIERARCHY.STATE	WebSphere MQ distributed publish/subscribe hierarchy relationship state

The attributes of the publish/subscribe system queues are shown in Table 10.

Table 10. Attributes of publish/subscribe system queues

Attribute	Default value
DEFPSIST	Yes
DEFSOPT	EXC
MAXMSGL	On AIX®, HP-UX, Linux, IBM i, Solaris, and Windows platforms: The value of the MAXMSGL parameter of the ALTER QMGR command
MAXDEPTH	999999999
SHARE	N/A
STGCLASS	This attribute is used only on z/OS platforms

Publish/subscribe system queue errors:

Errors can occur when distributed publish/subscribe queue manager queues are unavailable.

If the fan-out request queue SYSTEM.INTER.QMGR.FANREQ is unavailable, the MQSUB API receives reason codes and error messages written to the error log, on occasions where proxy subscriptions need to be delivered to directly connected queue managers.

If the hierarchy relationship state queue SYSTEM.HIERARCHY.STATE is unavailable, an error message is written to the error log and the publish/subscribe engine is put into COMPAT mode.

If any other of the SYSTEM.INTER.QMGR queues are unavailable, an error message is written to the error log, and although function is not disabled, it is likely that publish/subscribe messages will build up on queues on remote queue managers.

If the transmission queue to a parent, child or publish/subscribe cluster queue manager is unavailable:

1. The MQPUT API receives reason codes and the publications are not delivered.
2. Received inter-queue manager publications are backed out to the input queue, and subsequently re-attempted, being placed on the dead letter queue if the backout threshold is reached.
3. Proxy subscriptions are backed out to the fanout request queue, and subsequently attempted again, being placed on the dead letter queue if the backout threshold is reached; in which case the proxy subscription will not be delivered to any connected queue manager.
4. Hierarchy relationship protocol messages fail, and the connection status is marked as ERROR on the PUBSUB command.

Handling undelivered messages with the WebSphere MQ dead-letter queue handler

What is a dead-letter queue, how are messages put on it, and how do you manage it?

A *dead-letter queue* (DLQ), sometimes referred to as an *undelivered-message queue*, is a holding queue for messages that cannot be delivered to their destination queues. Every queue manager in a network should have an associated DLQ.

Messages can be put on the DLQ by queue managers, message channel agents (MCAs), and applications. All messages on the DLQ must be prefixed with a *dead-letter header* structure, MQDLH.

Messages put on the DLQ by a queue manager or a message channel agent always have an MQDLH; applications putting messages on the DLQ must supply an MQDLH. The *Reason* field of the MQDLH structure contains a reason code that identifies why the message is on the DLQ.

All WebSphere MQ environments need a routine to process messages on the DLQ regularly. WebSphere MQ supplies a default routine, called the *dead-letter queue handler* (the DLQ handler), which you invoke using the **runmqdlq** command.

Instructions for processing messages on the DLQ are supplied to the DLQ handler by means of a user-written *rules table*. That is, the DLQ handler matches messages on the DLQ against entries in the rules table; when a DLQ message matches an entry in the rules table, the DLQ handler performs the action associated with that entry.

Invoking the DLQ handler

Invoke the DLQ handler using the **runmqdlq** command. You can name the DLQ you want to process and the queue manager you want to use in two ways.

The two ways are as follows:

- As parameters to **runmqdlq** from the command prompt. For example:

  ```
  runmqdlq ABC1.DEAD.LETTER.QUEUE ABC1.QUEUE.MANAGER <qrule.rul
  ```

- In the rules table. For example:

  ```
  INPUTQ(ABC1.DEAD.LETTER.QUEUE) INPUTQM(ABC1.QUEUE.MANAGER)
  ```

The examples apply to the DLQ called ABC1.DEAD.LETTER.QUEUE, owned by the queue manager ABC1.QUEUE.MANAGER.

If you do not specify the DLQ or the queue manager as shown, the default queue manager for the installation is used along with the DLQ belonging to that queue manager.

The **runmqdlq** command takes its input from `stdin`; you associate the rules table with **runmqdlq** by redirecting `stdin` from the rules table.

To run the DLQ handler you must be authorized to access both the DLQ itself and any message queues to which messages on the DLQ are forwarded. For the DLQ handler to put messages on queues with the authority of the user ID in the message context, you must also be authorized to assume the identity of other users.

For more information about the **runmqdlq** command, see runmqdlq.

The sample DLQ handler, amqsdlq:

In addition to the DLQ handler invoked using the **runmqdlq** command, WebSphere MQ provides the source of a sample DLQ handler, amqsdlq, with a function that is similar to that provided by **runmqdlq**.

You can customize amqsdlq to provide a DLQ handler that meets your requirements. For example, you might decide that you want a DLQ handler that can process messages without dead-letter headers. (Both the default DLQ handler and the sample, amqsdlq, process only those messages on the DLQ that begin with a dead-letter header, MQDLH. Messages that do not begin with an MQDLH are identified as being in error, and remain on the DLQ indefinitely.)

MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.

In WebSphere MQ for Windows, the source of amqsdlq is supplied in the directory:

```
MQ_INSTALLATION_PATH\tools\c\samples\dlq
```

and the compiled version is supplied in the directory:

```
MQ_INSTALLATION_PATH\tools\c\samples\bin
```

In WebSphere MQ for UNIX and Linux systems, the source of amqsdlq is supplied in the directory:

```
MQ_INSTALLATION_PATH/samp/dlq
```

and the compiled version is supplied in the directory:

```
MQ_INSTALLATION_PATH/samp/bin
```

The DLQ handler rules table

The DLQ handler rules table defines how the DLQ handler is to process messages that arrive on the DLQ.

There are two types of entry in a rules table:
- The first entry in the table, which is optional, contains *control data*.
- All other entries in the table are *rules* for the DLQ handler to follow. Each rule consists of a *pattern* (a set of message characteristics) that a message is matched against, and an *action* to be taken when a message on the DLQ matches the specified pattern. There must be at least one rule in a rules table.

Each entry in the rules table comprises one or more keywords.

Control data:

This section describes the keywords that you can include in a control-data entry in a DLQ handler rules table.

Note:
- The vertical line (|) separates alternatives, only one of which can be specified.
- All keywords are optional.

INPUTQ (*QueueName***|'_')**

The name of the DLQ you want to process:
1. Any INPUTQ value you supply as a parameter to the **runmqdlq** command overrides any INPUTQ value in the rules table.
2. If you do not specify an INPUTQ value as a parameter to the **runmqdlq** command, but you **do** specify a value in the rules table, the INPUTQ value in the rules table is used.

3. If no DLQ is specified or you specify INPUTQ(' ') in the rules table, the name of the DLQ belonging to the queue manager with the name that is supplied as a parameter to the **runmqdlq** command is used.
4. If you do not specify an INPUTQ value as a parameter to the **runmqdlq** command or as a value in the rules table, the DLQ belonging to the queue manager named on the INPUTQM keyword in the rules table is used.

INPUTQM (*QueueManagerName*|'_')
The name of the queue manager that owns the DLQ named on the INPUTQ keyword:

1. Any INPUTQM value you supply as a parameter to the **runmqdlq** command overrides any INPUTQM value in the rules table.
2. If you do not specify an INPUTQM value as a parameter to the **runmqdlq** command, the INPUTQM value in the rules table is used.
3. If no queue manager is specified or you specify INPUTQM(' ') in the rules table, the default queue manager for the installation is used.

RETRYINT (*Interval*|60)
The interval, in seconds, at which the DLQ handler should reprocess messages on the DLQ that could not be processed at the first attempt, and for which repeated attempts have been requested. By default, the retry interval is 60 seconds.

WAIT (YES|NO|*nnn*)
Whether the DLQ handler should wait for further messages to arrive on the DLQ when it detects that there are no further messages that it can process.

YES The DLQ handler waits indefinitely.

NO The DLQ handler ends when it detects that the DLQ is either empty or contains no messages that it can process.

nnn The DLQ handler waits for *nnn* seconds for new work to arrive before ending, after it detects that the queue is either empty or contains no messages that it can process.

Specify WAIT (YES) for busy DLQs, and WAIT (NO) or WAIT (*nnn*) for DLQs that have a low level of activity. If the DLQ handler is allowed to terminate, invoke it again using triggering. For more information about triggering, see Starting WebSphere MQ applications using triggers.

An alternative to including control data in the rules table is to supply the names of the DLQ and its queue manager as input parameters to the **runmqdlq** command. If you specify a value both in the rules table and as input to the **runmqdlq** command, the value specified on the **runmqdlq** command takes precedence.

If you include a control-data entry in the rules table, it must be the **first** entry in the table.

Rules (patterns and actions):

A description of the pattern-matching keywords (those against which messages on the DLQ are matched), and the action keywords (those that determine how the DLQ handler is to process a matching message). An example rule is also provided.

The pattern-matching keywords

The pattern-matching keywords, which you use to specify values against which messages on the DLQ are matched, are as follows. (All pattern-matching keywords are optional):

APPLIDAT (*ApplIdentityData*|*)
The *ApplIdentityData* value specified in the message descriptor, MQMD, of the message on the DLQ.

APPLNAME (*PutApplName*|*)
The name of the application that issued the **MQPUT** or **MQPUT1** call, as specified in the *PutApplName* field of the message descriptor, MQMD, of the message on the DLQ.

APPLTYPE (*PutApplType*|*)
The *PutApplType* value, specified in the message descriptor, MQMD, of the message on the DLQ.

DESTQ (*QueueName*|*)
The name of the message queue for which the message is destined.

DESTQM (*QueueManagerName*|*)
The name of the queue manager of the message queue for which the message is destined.

FEEDBACK (*Feedback*|*)
When the *MsgType* value is MQFB_REPORT, *Feedback* describes the nature of the report.

You can use symbolic names. For example, you can use the symbolic name MQFB_COA to identify those messages on the DLQ that need confirmation of their arrival on their destination queues.

FORMAT (*Format*|*)
The name that the sender of the message uses to describe the format of the message data.

MSGTYPE (*MsgType*|*)
The message type of the message on the DLQ.

You can use symbolic names. For example, you can use the symbolic name MQMT_REQUEST to identify those messages on the DLQ that need replies.

PERSIST (*Persistence*|*)
The persistence value of the message. (The persistence of a message determines whether it survives restarts of the queue manager.)

You can use symbolic names. For example, you can use the symbolic name MQPER_PERSISTENT to identify messages on the DLQ that are persistent.

REASON (*ReasonCode*|*)
The reason code that describes why the message was put to the DLQ.

You can use symbolic names. For example, you can use the symbolic name MQRC_Q_FULL to identify those messages placed on the DLQ because their destination queues were full.

REPLYQ (*QueueName*|*)
The name of the reply-to queue specified in the message descriptor, MQMD, of the message on the DLQ.

REPLYQM (*QueueManagerName*|*)
The name of the queue manager of the reply-to queue, as specified in the message descriptor, MQMD, of the message on the DLQ.

USERID (*UserIdentifier*|*)
The user ID of the user who originated the message on the DLQ, as specified in the message descriptor, MQMD, of the message on the DLQ.

The action keywords

The action keywords, used to describe how a matching message is to be processed, are as follows:

ACTION (DISCARD|IGNORE|RETRY|FWD)
The action to be taken for any message on the DLQ that matches the pattern defined in this rule.

DISCARD
Delete the message from the DLQ.

IGNORE
Leave the message on the DLQ.

RETRY
If the first attempt to put the message on its destination queue fails, try again. The RETRY keyword sets the number of tries made to implement an action. The RETRYINT keyword of the control data controls the interval between attempts.

FWD Forward the message to the queue named on the FWDQ keyword.

You must specify the ACTION keyword.

FWDQ (*QueueName*|&DESTQ|&REPLYQ)
The name of the message queue to which to forward the message when ACTION (FWD) is requested.

QueueName
The name of a message queue. FWDQ(' ') is not valid.

&DESTQ
Take the queue name from the *DestQName* field in the MQDLH structure.

&REPLYQ
Take the queue name from the *ReplyToQ* field in the message descriptor, MQMD.

To avoid error messages when a rule specifying FWDQ (&REPLYQ) matches a message with a blank *ReplyToQ* field, specify REPLYQ (?*) in the message pattern.

FWDQM (*QueueManagerName*|&DESTQM|&REPLYQM|'_')
The queue manager of the queue to which to forward a message.

QueueManagerName
The name of the queue manager of the queue to which to forward a message when ACTION (FWD) is requested.

&DESTQM
Take the queue manager name from the *DestQMgrName* field in the MQDLH structure.

&REPLYQM
Take the queue manager name from the *ReplyToQMgr* field in the message descriptor, MQMD.

' ' FWDQM(' '), which is the default value, identifies the local queue manager.

HEADER (YES|NO)
Whether the MQDLH should remain on a message for which ACTION (FWD) is requested. By default, the MQDLH remains on the message. The HEADER keyword is not valid for actions other than FWD.

PUTAUT (DEF|CTX)
The authority with which messages should be put by the DLQ handler:

DEF Put messages with the authority of the DLQ handler itself.

CTX Put the messages with the authority of the user ID in the message context. If you specify PUTAUT (CTX), you must be authorized to assume the identity of other users.

RETRY (*RetryCount*|1)
The number of times, in the range 1 - 999,999,999, to try an action (at the interval specified on the RETRYINT keyword of the control data). The count of attempts made by the DLQ handler to implement any particular rule is specific to the current instance of the DLQ handler; the count does not persist across restarts. If the DLQ handler is restarted, the count of attempts made to apply a rule is reset to zero.

Example rule

Here is an example rule from a DLQ handler rules table:

```
PERSIST(MQPER_PERSISTENT) REASON (MQRC_PUT_INHIBITED) +
 ACTION (RETRY) RETRY (3)
```

This rule instructs the DLQ handler to make three attempts to deliver to its destination queue any persistent message that was put on the DLQ because **MQPUT** and **MQPUT1** were inhibited.

All keywords that you can use on a rule are described in the rest of this section. Note the following:
- The default value for a keyword, if any, is underlined. For most keywords, the default value is * (asterisk), which matches any value.
- The vertical line (|) separates alternatives, only one of which can be specified.
- All keywords except ACTION are optional.

Rules table conventions:

The syntax, structure and contents of the DLQ handler rules table must adhere to these conventions.

The rules table must adhere to the following conventions:
- A rules table must contain at least one rule.
- Keywords can occur in any order.
- A keyword can be included only once in any rule.
- Keywords are not case-sensitive.
- A keyword and its parameter value must be separated from other keywords by at least one blank or comma.
- There can be any number of blanks at the beginning or end of a rule, and between keywords, punctuation, and values.
- Each rule must begin on a new line.
- On Windows systems, the last rule in the table must end with a carriage return/line feed character. You can achieve this by ensuring that you press the Enter key at the end of the rule, so that the last line of the table is a blank line.
- For reasons of portability, the significant length of a line must not be greater than 72 characters.
- Use the plus sign (+) as the last nonblank character on a line to indicate that the rule continues from the first nonblank character in the next line. Use the minus sign (-) as the last nonblank character on a line to indicate that the rule continues from the start of the next line. Continuation characters can occur within keywords and parameters.

  For example:

  ```
  APPLNAME('ABC+
    D')
  ```

  results in 'ABCD', and

  ```
  APPLNAME('ABC-
    D')
  ```

  results in 'ABC D'.
- Comment lines, which begin with an asterisk (*), can occur anywhere in the rules table.
- Blank lines are ignored.
- Each entry in the DLQ handler rules table comprises one or more keywords and their associated parameters. The parameters must follow these syntax rules:
  - Each parameter value must include at least one significant character. The delimiting single quotation marks in values that are enclosed in quotation marks are not considered to be significant. For example, these parameters are valid:

`FORMAT('ABC')`	3 significant characters
`FORMAT(ABC)`	3 significant characters
`FORMAT('A')`	1 significant character
`FORMAT(A)`	1 significant character
`FORMAT(' ')`	1 significant character

These parameters are invalid because they contain no significant characters:

```
FORMAT('')
FORMAT( )
FORMAT()
FORMAT
```

– Wildcard characters are supported. You can use the question mark (?) instead of any single character, except a trailing blank; you can use the asterisk (*) instead of zero or more adjacent characters. The asterisk (*) and the question mark (?) are ***always*** interpreted as wildcard characters in parameter values.

– Wildcard characters cannot be included in the parameters of these keywords: ACTION, HEADER, RETRY, FWDQ, FWDQM, and PUTAUT.

– Trailing blanks in parameter values, and in the corresponding fields in the message on the DLQ, are not significant when performing wildcard matches. However, leading and embedded blanks within strings that are enclosed in single quotation marks are significant to wildcard matches.

– Numeric parameters cannot include the question mark (?) wildcard character. You can use the asterisk (*) instead of an entire numeric parameter, but not as part of a numeric parameter. For example, these are valid numeric parameters:

`MSGTYPE(2)`	Only reply messages are eligible
`MSGTYPE(*)`	Any message type is eligible
`MSGTYPE('*')`	Any message type is eligible

However, `MSGTYPE('2*')` is not valid, because it includes an asterisk (*) as part of a numeric parameter.

– Numeric parameters must be in the range 0-999 999 999. If the parameter value is in this range, it is accepted, even if it is not currently valid in the field to which the keyword relates. You can use symbolic names for numeric parameters.

– If a string value is shorter than the field in the MQDLH or MQMD to which the keyword relates, the value is padded with blanks to the length of the field. If the value, excluding asterisks, is longer than the field, an error is diagnosed. For example, these are all valid string values for an 8 character field:

`'ABCDEFGH'`	8 characters
`'A*C*E*G*I'`	5 characters excluding asterisks
`'*A*C*E*G*I*K*M*O*'`	8 characters excluding asterisks

– Enclose strings that contain blanks, lowercase characters, or special characters other than period (.), forward slash (?), underscore (_), and percent sign (%) in single quotation marks. Lowercase characters not enclosed in single quotation marks are folded to uppercase. If the string includes a quotation, use two single quotation marks to denote both the beginning and the end of the quotation. When the length of the string is calculated, each occurrence of double quotation marks is counted as a single character.

How the rules table is processed

The DLQ handler searches the rules table for a rule where the pattern matches a message on the DLQ.

The search begins with the first rule in the table, and continues sequentially through the table. When the DLQ handler finds a rule with a matching pattern, it takes the action from that rule. The DLQ handler increments the retry count for a rule by 1 whenever it applies that rule. If the first try fails, the DLQ handler tries again until the number of tries matches the number specified on the RETRY keyword. If all attempts fail, the DLQ handler searches for the next matching rule in the table.

This process is repeated for subsequent matching rules until an action is successful. When each matching rule has been attempted the number of times specified on its RETRY keyword, and all attempts have failed, ACTION (IGNORE) is assumed. ACTION (IGNORE) is also assumed if no matching rule is found.

Note:

1. Matching rule patterns are sought only for messages on the DLQ that begin with an MQDLH. Messages that do not begin with an MQDLH are reported periodically as being in error, and remain on the DLQ indefinitely.
2. All pattern keywords can be allowed to default, such that a rule can consist of an action only. Note, however, that action-only rules are applied to all messages on the queue that have MQDLHs and that have not already been processed in accordance with other rules in the table.
3. The rules table is validated when the DLQ handler starts, and errors are flagged at that time. You can make changes to the rules table at any time, but those changes do not come into effect until the DLQ handler restarts.
4. The DLQ handler does not alter the content of messages, the MQDLH, or the message descriptor. The DLQ handler always puts messages to other queues with the message option MQPMO_PASS_ALL_CONTEXT.
5. Consecutive syntax errors in the rules table might not be recognized because the rules table is designed to eliminate the generation of repetitive errors during validation.
6. The DLQ handler opens the DLQ with the MQOO_INPUT_AS_Q_DEF option.
7. Multiple instances of the DLQ handler can run concurrently against the same queue, using the same rules table. However, it is more usual for there to be a one-to-one relationship between a DLQ and a DLQ handler.

Ensuring that all DLQ messages are processed:

The DLQ handler keeps a record of all messages on the DLQ that have been seen but not removed.

If you use the DLQ handler as a filter to extract a small subset of the messages from the DLQ, the DLQ handler still has to keep a record of those messages on the DLQ that it did not process. Also, the DLQ handler cannot guarantee that new messages arriving on the DLQ are seen, even if the DLQ is defined as first-in-first-out (FIFO). If the queue is not empty, the DLQ is periodically re-scanned to check all messages.

For these reasons, try to ensure that the DLQ contains as few messages as possible; if messages that cannot be discarded or forwarded to other queues (for whatever reason) are allowed to accumulate on the queue, the workload of the DLQ handler increases and the DLQ itself can fill up.

You can take specific measures to enable the DLQ handler to empty the DLQ. For example, try not to use ACTION (IGNORE), which leaves messages on the DLQ. (Remember that ACTION (IGNORE) is assumed for messages that are not explicitly addressed by other rules in the table.) Instead, for those messages that you would otherwise ignore, use an action that moves the messages to another queue. For example:

```
ACTION (FWD) FWDQ (IGNORED.DEAD.QUEUE) HEADER (YES)
```

Similarly, make the final rule in the table a catchall to process messages that have not been addressed by earlier rules in the table. For example, the final rule in the table could be something like this:

```
 ACTION (FWD) FWDQ (REALLY.DEAD.QUEUE) HEADER (YES)
```

This forwards messages that fall through to the final rule in the table to the queue REALLY.DEAD.QUEUE, where they can be processed manually. If you do not have such a rule, messages are likely to remain on the DLQ indefinitely.

An example DLQ handler rules table

An example rules table for the runmqdlq command, containing a single control-data entry and several rules.

```
*************************************************************************
*       An example rules table for the runmqdlq command                *
*************************************************************************
* Control data entry
* ------------------
* If no queue manager name is supplied as an explicit parameter to
* runmqdlq, use the default queue manager for the machine.
* If no queue name is supplied as an explicit parameter to runmqdlq,
* use the DLQ defined for the local queue manager.
*
inputqm(' ')  inputq(' ')

* Rules
* -----
* We include rules with ACTION (RETRY) first to try to
* deliver the message to the intended destination.
* If a message is placed on the DLQ because its destination
* queue is full, attempt to forward the message to its
* destination queue. Make 5 attempts at approximately
* 60-second intervals (the default value for RETRYINT).

REASON(MQRC_Q_FULL) ACTION(RETRY) RETRY(5)

* If a message is placed on the DLQ because of a put inhibited
* condition, attempt to forward the message to its
* destination queue. Make 5 attempts at approximately
* 60-second intervals (the default value for RETRYINT).

REASON(MQRC_PUT_INHIBITED) ACTION(RETRY) RETRY(5)

* The AAAA corporation are always sending messages with incorrect
* addresses. When we find a request from the AAAA corporation,
* we return it to the DLQ (DEADQ) of the reply-to queue manager
* (&REPLYQM).
* The AAAA DLQ handler attempts to redirect the message.

MSGTYPE(MQMT_REQUEST) REPLYQM(AAAA.*) +
  ACTION(FWD) FWDQ(DEADQ) FWDQM(&REPLYQM)

* The BBBB corporation never do things by half measures. If
* the queue manager BBBB.1 is unavailable, try to
* send the message to BBBB.2

DESTQM(bbbb.1) +
  action(fwd) fwdq(&DESTQ) fwdqm(bbbb.2) header(no)

* The CCCC corporation considers itself very security
* conscious, and believes that none of its messages
* will ever end up on one of our DLQs.
* Whenever we see a message from a CCCC queue manager on our
* DLQ, we send it to a special destination in the CCCC organization
* where the problem is investigated.

REPLYQM(CCCC.*) +
  ACTION(FWD) FWDQ(ALARM) FWDQM(CCCC.SYSTEM)
* Messages that are not persistent run the risk of being
* lost when a queue manager terminates. If an application
* is sending nonpersistent messages, it should be able
* to cope with the message being lost, so we can afford to
* discard the message.   PERSIST(MQPER_NOT_PERSISTENT) ACTION(DISCARD)
* For performance and efficiency reasons, we like to keep
* the number of messages on the DLQ small.
* If we receive a message that has not been processed by
* an earlier rule in the table, we assume that it
* requires manual intervention to resolve the problem.
* Some problems are best solved at the node where the
* problem was detected, and others are best solved where
* the message originated. We don't have the message origin,
* but we can use the REPLYQM to identify a node that has
* some interest in this message.
* Attempt to put the message onto a manual intervention
* queue at the appropriate node. If this fails,
* put the message on the manual intervention queue at
* this node.

REPLYQM('?*') +
  ACTION(FWD) FWDQ(DEADQ.MANUAL.INTERVENTION) FWDQM(&REPLYQM)

ACTION(FWD) FWDQ(DEADQ.MANUAL.INTERVENTION)
```

Multiple installations

On , it is possible to have more than one copy of on a system.

You can choose where each copy of is installed, but each copy must be in a separate installation location. A maximum of 128 installations of can exist on a single machine at a time. One installation can be an installation of at fix pack 6, or later. You now have a choice:

- Keep the simplicity of maintaining and managing a single installation of on a machine.
- Take advantage of the flexibility offered by enabling multiple installations.

Before you install multiple copies of , you must make several decisions:

- Will you have a copy of on the system?

  When at fix pack 6, or later, is installed on the system, there are a number of restrictions to consider:
  - On systems, must be installed in the default location.
  - must be the first installation on a system. You cannot install after installing version 7.1, or later. If you uninstall version 7.0.1, it cannot be reinstalled while a later version of WebSphere MQ is installed.
  - is automatically the primary installation. You cannot select another installation as the primary installation while is installed.
- Where will you install each copy of ?

  You can choose the installation location for your installations at version 7.1, or later. For more information, see ../../../com.ibm.mq.ins.doc/q008310_.dita.
- Do you need a primary installation?

  A primary installation is an installation to which system-wide locations refer. For more information, see ../../../com.ibm.mq.ins.doc/q005740_.dita.
- How will your applications connect?

You need to consider how your applications locate the appropriate libraries. For more information, see Connecting applications in a multiple installation environment, and Connecting .NET applications in a multiple installation environment.

- Do your existing exits need changing?

  If is not installed in the default location, your exits need to be updated. For more information, see Writing and compiling exits and installable services .

- Which queue manager will be associated with which installation?

  Each queue manager is associated with a particular installation. The installation that a queue manager is associated with limits that queue manager so that it can be administered only by commands from that installation. For more information, see Associating a queue manager with an installation.

- How will you set up your environment to work with each installation?

  With multiple installations on a system, you need to consider how you will work with particular installations, and how you will issue commands from that installation. You can either specify the full path to the command, or you can use the **setmqenv** or **crtmqenv** command to set environment variables. Setting the environment variables allows you to omit the path to the commands for that installation. For more information, see setmqenv, and crtmqenv.

When you have answered these questions, you can install using the steps provided in ../../../com.ibm.mq.ins.doc/q008580_.dita.

If you have existing installations of and you want to use the multiple installation capability to migrate from one version of to another version, see Multi-installation queue manager coexistence on , , and .

The IBM message service client for .NET support pack and multiple installations

For multiple version support, the *Java and .NET Messaging and Web Services* feature must be installed with the product. This feature contains all the functionality included in the *IBM Message Service Client for .NET* support pack (IA9H). If the support pack is installed on the system, multiple versions are not supported. You must uninstall the support pack before installing . For more information about installing the .NET feature, see Installing WebSphere MQ classes for .NET.

Related information:

Configuring multiple installations

Finding installations of WebSphere MQ on a system

UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1 to version 7.5

UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1 to version 7.5

Choosing a primary installation

On systems that support multiple installations of (), the primary installation is the one to which system-wide locations refer. Having a primary installation is optional, but convenient.

Before , only one instance of the product could be installed at any time. On systems, several global environment variables were set to point to that installation. On systems, symbolic links were added to `/usr/lib`, `/usr/bin`, and `/usr/include`, also pointing at that single installation.

From , you can install multiple versions of on . It is possible to have more than one installation of on one of these systems at any time and, optionally, to configure one of these installations as the primary installation. Environment variables and symbolic links pointing to a single installation are less meaningful when multiple versions exist. However, some functions require these system-wide locations to work. For example, custom user scripts for administering , and third party products. These functions work only on the primary installation.

On systems, if you set an installation as the primary installation, symbolic links to the external libraries and control commands of that installation are added into `/usr/lib`, and `/usr/bin`. If you do not have a primary installation, the symbolic links are not created. For a list of the symbolic links that are made to the primary installation, see ../../../com.ibm.mq.ins.doc/q008280_.dita.

On systems, the global environmental variables point to the directories into which the primary installation was installed. These environment variables are used to locate libraries, control commands, and header files. Additionally, on systems, some features of the operating system require the central registration of interface libraries that are then loaded into a single process. With multiple versions of , there would be conflicting sets of libraries. The features would try to load these conflicting sets of libraries into a single process. Therefore, such features can be used only with the primary installation. For details about some of the features that are limited to use with the primary installation, see ../../../com.ibm.mq.ins.doc/q008290_.dita.

If you have an installation of on the system, this installation is automatically the primary installation. The primary installation cannot be changed while is installed. If all the installations on the system are at , or later, you can choose whether to have a primary installation. Consider the options in ../../../com.ibm.mq.ins.doc/q005740_.dita#q005740_/PrimaryInstallationOptions.

Table 11. Primary installation options.

This table shows the valid installation configurations for primary installations. With a single , or later, it can be either primary, or non-primary. With multiple installations, one at and one or more at , or later, must be the primary, and the other installations must be non-primary. With multiple installations at , or later, one installation can be primary, or all installations can be non-primary.

Options	**Valid installation configurations**		**More information**
	Primary	Non-primary	
Single installation of , or later.	, or later.	None	If you want to continue working with a single installation in the same way as previous releases, configure your installation as the primary installation. For information about this option, see Single installation of , or later, configured as the primary installation
	None	, or later.	If you want to continue working with a single installation, but do not want symbolic links or global environment variables created for you, configure your installation as non-primary. For information about the implications of this option, see Single installation of , or later, configured as non-primary
Multiple installations: and , or later.		, or later.	If you want to have multiple installations of , with one at version 7.0.1, the version 7.0.1 installation is automatically the primary installation. While version 7.0.1 is installed, you cannot change which installation is the primary installation. For information about this option and its implications, see Multiple installations of , one at
Multiple installations: , or later.	, or later.	, or later.	If you want to have multiple installations of WebSphere MQ at version 7.1 or greater, you can choose whether to make one of the installations primary. For information about this option, see Multiple installations of , or later
	None	, or later.	

Related information:

Single installation of WebSphere MQ Version 7.1, or later, configured as the primary installation

Single installation of WebSphere MQ Version 7.1, or later, configured as non-primary

Multiple installations of WebSphere MQ Version 7.1, or later

Multiple installations of WebSphere MQ, one at version 7.0.1

Changing the primary installation

../../../com.ibm.mq.ins.doc/q008310_.dita

../../../com.ibm.mq.ins.doc/q008260_.dita

../../../com.ibm.mq.ins.doc/q008270_.dita

Single installation of IBM WebSphere MQ Version 7.1, or later, configured as the primary installation

Marking a IBM WebSphere MQ installation as primary adds symbolic links, or global environment variables to the system so that the IBM WebSphere MQ commands and libraries used by applications are automatically available with minimum system setup required.

You decide where to install IBM WebSphere MQ.

Where possible, configure applications and scripts to use the system search path to find the IBM WebSphere MQ control commands or IBM WebSphere MQ libraries. This configuration of applications and scripts provides maximum flexibility for undertaking future tasks such as migrating to the next release of IBM WebSphere MQ, or installing a second installation. For more information about options for connecting your applications, see Connecting applications in a multiple installation environment.

On Windows, the first installation is automatically configured as the primary installation. On UNIX and Linux platforms, the first installation onto a system must be manually configured to be the primary installation. Set the primary installation using the **setmqinst** command. For more information, see Uninstalling, upgrading, and maintaining the primary installation.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Choosing an installation location
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

Planning your installation
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

Single installation of IBM WebSphere MQ Version 7.1, or later, configured as non-primary

If you install IBM WebSphere MQ Version 7.1, or later, as non-primary you might have to configure a library path for applications to load IBM WebSphere MQ libraries. On Windows, some product capabilities are available only when IBM WebSphere MQ is configured as primary.

UNIX and Linux systems

The implications of running a non-primary installation on UNIX and Linux are:

- Applications that locate their IBM WebSphere MQ libraries using an embedded library path, for example, RPATH, cannot find those libraries if the following conditions are true:
  - IBM WebSphere MQ is installed in a different directory from the directory specified in the RPATH
  - There are no symbolic links in /usr
- Where applications locate their libraries using an external library path, for example, LD_LIBRARY_PATH, you must configure the external library path to include the *MQ_INSTALLATION_PATH*/lib or *MQ_INSTALLATION_PATH*/lib64 directory. The **setmqenv** and **crtmqenv** commands can configure a number of environment variables in the current shell, including the external library path.
- Most IBM WebSphere MQ processes run as setuid/setgid. As a result, when loading user exits they ignore the external library path. User exits that reference IBM WebSphere MQ libraries can find those libraries only if they are found in the library path embedded within them. They would be resolved if there were a symbolic link in /usr. User exits that are intended to be run on IBM WebSphere MQ Version 7.1, or later can now be built so that they do not refer to IBM WebSphere MQ libraries at all. Instead they rely on IBM WebSphere MQ to pass in function pointers to the IBM WebSphere MQ functions that the exit can then use. For more information, see Writing and compiling exits and installable services.

For more information about options for connecting your applications, see Connecting applications in a multiple installation environment.

On UNIX and Linux platforms, the first installation onto a system is not automatically configured as the primary installation. However, a single symbolic link is included in /usr/bin to locate the **dspmqver** command. If you do not want any symbolic links, you must remove this link using the following command:

```
setmqinst -x -p MQ_INSTALLATION_PATH
```

Windows systems

The implications of running a non-primary installation on Windows are:

- Applications normally find their libraries using the external library path, PATH. There is no concept of an embedded library path or explicit library location. If the installation is non-primary, the global PATH environment variable does not contain the IBM WebSphere MQ installation directory. For applications to find IBM WebSphere MQ libraries, update the PATH environment variable to reference the IBM WebSphere MQ installation directory. The **setmqenv** and **crtmqenv** commands can configure a number of environment variables in the current shell, including the external library path.
- Some product capabilities are available only when an installation is configured as the primary installation; see Features that can be used only with the primary installation on Windows.

By default, on Windows, the first installation is automatically configured as primary. You must manually deselect it as the primary installation.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Choosing an installation location
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

Planning your installation
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

setmqenv

crtmqenv

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

Multiple installations of IBM WebSphere MQ Version 7.1, or later

You can choose to have one of the IBM WebSphere MQ Version 7.1 installations configured as the primary installation. Your choice depends on how applications locate libraries.

The IBM WebSphere MQ libraries, such as mqm, which ship with IBM WebSphere MQ Version 7.1 automatically use libraries of the level required by the queue manager to which they are connecting. This means that provided an application locates its IBM WebSphere MQ libraries from a IBM WebSphere MQ Version 7.1 installation, it can connect to any queue manager on that system. Having one IBM WebSphere MQ Version 7.1 installation configured as primary ensures that if the application finds its IBM WebSphere MQ interface library, the application can connect to any queue manager.

For more information about connecting applications in a multiple installation environment, see Connecting applications in a multiple installation environment.

The primary installation is not automatically changed when you uninstall the primary installation. If you want another installation to be the primary installation, you must manually set the primary installation using the **setmqinst** command. For more information, see Uninstalling, upgrading, and maintaining the primary installation.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Choosing an installation location
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

Multiple installations
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

Planning your installation
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

Multiple installations of IBM WebSphere MQ, one at Version 7.0.1

IBM WebSphere MQ Version 7.1, or later, can coexist with IBM WebSphere MQ Version 7.0.1 with some limitations.

- On UNIX and Linux systems, Version 7.0.1 can be installed only in a fixed, default location, so you cannot install Version 7.1, or later, in that default location.
- IBM WebSphere MQ Version 7.0.1 is automatically configured as the primary installation. On UNIX and Linux systems, symbolic links are automatically created to the appropriate IBM WebSphere MQ directories. On Windows, everything that the product provided is registered globally. IBM WebSphere MQ Version 7.0.1 must be installed in this way to work. So where IBM WebSphere MQ Version 7.0.1 is installed, a IBM WebSphere MQ Version 7.1, or later, installation cannot be made primary.

The libraries from IBM WebSphere MQ Version 7.1, or later, can work with any queue manager running under IBM WebSphere MQ Version 7.0.1, or later. If an application needs to connect to queue managers running under Version 7.0.1 as well as later versions, it can continue to operate normally if the following conditions are true:

- It locates IBM WebSphere MQ Version 7.1, or later, libraries at run time.
- It uses only functions available in Version 7.0.1.

For more information about connecting applications in a multiple installation environment, see Connecting applications in a multiple installation environment.

The primary installation is not automatically changed when you uninstall IBM WebSphere MQ Version 7.0.1. If you want another installation to be the primary installation, you must manually set the primary installation using the **setmqinst** command. For more information, see Uninstalling, upgrading, and maintaining the primary installation.

Related information:

Choosing an installation location
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

Planning your installation
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

Multiple installations
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

Planning your storage and performance requirements

You must set realistic and achievable storage, and performance goals for your IBM WebSphere MQ system. Use the links to find out about factors that affect storage and performance on your platform.

The requirements vary depending on the systems that you are using IBM WebSphere MQ on, and what components you want to use.

For the latest information about supported hardware and software environments, see the IBM WebSphere MQ requirements Web site:

www.ibm.com/software/integration/wmq/requirements/

IBM WebSphere MQ stores queue manager data in the file system. Use the following links to find out about planning and configuring directory structures for use with IBM WebSphere MQ:

- “Planning file system support” on page 119
- “Requirements for shared file systems” on page 120
- “Sharing WebSphere MQ files” on page 128
- “Directory structure on UNIX and Linux systems” on page 131
- “Directory structure on Windows systems” on page 140

Use the following links for information about system resources, shared memory, and process priority on UNIX and Linux:

- “WebSphere MQ and UNIX System V IPC resources” on page 143

- “Shared memory on AIX” on page 144
- “WebSphere MQ and UNIX Process Priority” on page 144

Related concepts:

“Planning” on page 1
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

“Designing a IBM WebSphere MQ architecture” on page 1
Find out about the different architectures that IBM WebSphere MQ supports for point-to-point and publish/subscribe messaging styles.

Related information:

Hardware and software requirements on UNIX and Linux
Before you install IBM WebSphere MQ, check that your system meets the hardware and operating system software requirements for the particular components you intend to install.

Hardware and software requirements on Windows
Check that the server environment meets the prerequisites for installing IBM WebSphere MQ for Windows and install any prerequisite software that is missing from your system from the server DVD.

Disk space requirements

The storage requirements for WebSphere MQ depend on which components you install, and how much working space you need.

Disk storage is required for the optional components you choose to install, including any prerequisite components they require. The total storage requirement also depends on the number of queues that you use, the number and size of the messages on the queues, and whether the messages are persistent. You also require archiving capacity on disk, tape, or other media, as well as space for your own application programs.

The following table shows the approximate disk space required when you install various combinations of the product on different platforms. (Values are rounded up to the nearest 5 MB, where a MB is 1,048,576 bytes.)

Platform	Client installation [1]	Server installation [2]	WebSphere MQ MFT installation [3]	Full installation [4]
AIX	145 MB	190 MB	705 MB	915 MB
HP-UX	225 MB	310 MB	1075 MB	1340 MB
IBM i	215 MB	450 MB	80 MB	655 MB
Linux for System x (32 bit)	85 MB	N/A	N/A	120 MB
Linux for System x (64 bit)	125 MB	170 MB	575 MB	935 MB
Linux on POWER Systems - Big Endian	130 MB	170 MB	565 MB	715 MB
Solaris x86-64, AMD64, EM64T, and compatible processors	105 MB [5]	150 MB [5]	695 MB	860 MB
Solaris SPARC	105 MB [5]	150 MB [5]	680 MB	820 MB
Windows (32 bit install) [6]	390 MB	N/A	N/A	475 MB
Windows (64 bit install) [6]	445 MB	555 MB	710 MB	1005 MB

Usage notes

1. A client installation includes the following components:
   - Runtime
   - Client
2. A server installation includes the following components:
   - Runtime
   - Server
3. An WebSphere MQ Managed File Transfer installation includes the following components:
   - WebSphere MQ Managed File Transfer Service, Logger, Agent, Tools, and Base components
   - Runtime
   - Server
   - Java™
   - JRE
4. A full installation includes all available components.
5. Solaris On Solaris platforms you must install silently to get this combination of components.
6. Windows Not all the components listed here are installable features on Windows systems; their functionality is sometimes included in other features. See WebSphere MQ features for Windows systems.

Related information:

Choosing what to install
You can select the components or features that you require when you install IBM WebSphere MQ.

Planning file system support

Queue manager data is stored in the file system. A queue manager makes use of file system locking to prevent multiple instances of a multi-instance queue manager being active at the same time.

Shared file systems

Shared file systems enable multiple systems to access the same physical storage device concurrently. Corruption would occur if multiple systems accessed the same physical storage device directly without some means of enforcing locking and concurrency control. Operating systems provide local file systems with locking and concurrency control for local processes; network file systems provide locking and concurrency control for distributed systems.

Historically, networked file systems have not performed fast enough, or provided sufficient locking and concurrency control, to meet the requirements for logging messages. Today, networked file systems can provide good performance, and implementations of reliable network file system protocols such as *RFC 3530, Network File System (NFS) version 4 protocol*, meet the requirements for logging messages reliably.

Shared file systems and WebSphere MQ

Queue manager data for a multi-instance queue manager is stored in a shared network file system. On Microsoft Windows, UNIX and Linux systems, the queue manager's data files and log files must be placed in shared network file system.

Prior to release v7.0.1, WebSphere MQ does not support queue manager data stored on networked storage accessed as a shared file system. If queue manager data is placed on shared networked storage, then you need to ensure the queue manager data is not accessed by another instance of the queue manager running at the same time.

From v7.0.1 onwards, WebSphere MQ uses locking to prevent multiple instances of the same multi-instance queue manager being active at the same time. The same locking also ensures that two separate queue managers cannot inadvertently use the same set of queue manager data files. Only one instance of a queue manager can have its lock at a time. Consequently, WebSphere MQ does support queue manager data stored on networked storage accessed as a shared file system.

Because not all the locking protocols of network file systems are robust, and because a file system might be configured for performance rather than data integrity, you must run the **amqmfsck** command to test whether a network file system will control access to queue manager data and logs correctly. This command applies only UNIX and IBM i systems. On Microsoft Windows, there is only one supported network file system and the **amqmfsck** command is not required.

Related tasks:

"Verifying shared file system behavior" on page 121
Run **amqmfsck** to check whether a shared file system on UNIX systems meets the requirements for storing the queue manager data of a multi-instance queue manager. Run the IBM WebSphere MQ MQI client sample program **amqsfhac** in parallel with **amqmfsck** to demonstrate that a queue manager maintains message integrity during a failure.

Requirements for shared file systems

Shared files systems must provide data write integrity, guaranteed exclusive access to files and release locks on failure to work reliably with WebSphere MQ.

There are three fundamental requirements that a shared file system must meet to log messages reliably:

1. Data write integrity.

   Data write integrity is sometimes called "Write through to disk on flush". The queue manager must be able to synchronize with data being successfully committed to the physical device. In a transactional system, you need to be sure that some writes have been safely committed before continuing with other processing.

   More specifically, WebSphere MQ on UNIX platforms uses the *O_SYNC* open option and the `fsync()` system call to explicitly force writes to recoverable media, and is dependent upon these options operating correctly.

   Attention: Linux You should mount the file system with the `async` option, which still supports the option of synchronous writes and gives better performance than the `sync` option.

   Note, however, that if the file system has been exported from Linux, you must still export the file system using the `sync` option.
2. Guaranteed exclusive access to files.

   In order to synchronize multiple queue managers, there needs to be a mechanism for a queue manager to obtain an exclusive lock on a file.
3. Release locks on failure.

   If a queue manager fails, or if there is a communication failure with the file system, files locked by the queue manager need to be unlocked and made available to other processes without waiting for the queue manager to be reconnected to the file system.

   Modern file systems, such as Network File System (NFS) Version 4, use leased locks to detect failures and then release locks following a failure. Older file systems such as Network File System Version 3, which do not have a reliable mechanism to release locks after a failure, must not be used with multi-instance queue managers.

A shared file system *must* meet these requirements for WebSphere MQ to operate reliably. If it does not, the queue manager data and logs get corrupted when using the shared file system in a multi-instance queue manager configuration.

You must check whether the shared file system you plan to use meets these requirements. You must also check whether the file system is correctly configured for reliability. Shared file systems sometimes provide configuration options to improve performance at the expense of reliability.

Under normal circumstances WebSphere MQ operates correctly with attribute caching and it is not necessary to disable caching, for example by setting NOAC on an NFS mount. Attribute caching can cause issues when multiple file system clients are contending for write access to the same file on the file system server, as the cached attributes used by each client might not be the same as those attributes on the server. An example of files accessed in this way are queue manager error logs for a multi-instance queue manager. The queue manager error logs might be written to by both an active and a standby queue manager instance and cached file attributes might cause the error logs to grow larger than expected, before rollover of the files occurs.

To help to check the file system, run the task Verifying shared file system behavior. This task checks if your shared file system meets requirements 2 and 3. You need to verify requirement 1 in your shared file system documentation, or by experimenting with logging data to the disk.

Disk faults can cause errors when writing to disk, which WebSphere MQ reports as First Failure Data Capture errors. You can run the file system checker for your operating system to check the shared file system for any disk faults. For example, on UNIX and Linux platforms the file system checker is called fsck. On Windows platforms the file system checker is called CHKDSK, or SCANDISK.

Note: You should put only queue manager data on an NFS server. On the NFS, use the following three options with the mount command to make the system secure:

noexec
: By using this option, you stop binary files from being run on the NFS, which prevents a remote user from running unwanted code on the system.

nosuid
: By using this option, you prevent the use of the set-user-identifier and set-group-identifier bits, which prevents a remote user from gaining higher privileges.

nodev
: By using this option, you stop character and block special devices from being used or defined, which prevents a remote user from getting out of a chroot jail.

Verifying shared file system behavior:

Run **`amqmfsck`** to check whether a shared file system on UNIX systems meets the requirements for storing the queue manager data of a multi-instance queue manager. Run the IBM WebSphere MQ MQI client sample program **`amqsfhac`** in parallel with **`amqmfsck`** to demonstrate that a queue manager maintains message integrity during a failure.

Before you begin

You need a server with networked storage, and two other servers connected to it that have WebSphere MQ installed. You must have administrator (root) authority to configure the file system, and be a WebSphere MQ Administrator to run **`amqmfsck`**.

About this task

"Requirements for shared file systems" on page 120 describes the file system requirements for using a shared file system with multi-instance queue managers. The IBM WebSphere MQ technote Testing and support statement for WebSphere MQ multi-instance queue managers lists the shared file systems that IBM has already tested with. The procedure in this task describes how to test a file system to help you assess whether an unlisted file system maintains data integrity.

Failover of a multi-instance queue manager can be triggered by hardware or software failures, including networking problems which prevent the queue manager writing to its data or log files. Mainly, you are interested in causing failures on the file server. But you must also cause the IBM WebSphere MQ servers to fail, to test any locks are successfully released. To be confident in a shared file system, test all of the following failures, and any other failures that are specific to your environment:

1. Shutting down the operating system on the file server including syncing the disks.
2. Halting the operating system on the file server without syncing the disks.
3. Pressing the reset button on each of the servers.
4. Pulling the network cable out of each of the servers.
5. Pulling the power cable out of each of the servers.
6. Switching off each of the servers.

Create the directory on the networked storage that you are going to use to share queue manager data and logs. The directory owner must be a WebSphere MQ Administrator, or in other words, a member of the `mqm` group on UNIX. The user who runs the tests must have WebSphere MQ Administrator authority.

Use the example of exporting and mounting a file system in Create a multi-instance queue manager on Linux to help you through configuring the file system. Different file systems require different configuration steps. Read the file system documentation.

Procedure

In each of the checks, cause all the failures in the previous list while the file system checker is running. If you intend to run **amqsfhac** at the same time as **amqmfsck**, do the task, “Running **amqsfhac** to test message integrity” on page 126 in parallel with this task.

1. Mount the exported directory on the two IBM WebSphere MQ servers.

   On the file system server create a shared directory `shared`, and a subdirectory to save the data for multi-instance queue managers, `qmdata`. For an example of setting up a shared directory for multi-instance queue managers on Linux, see Example in Create a multi-instance queue manager on Linux
2. Check basic file system behavior.

   On one IBM WebSphere MQ server, run the file system checker with no parameters.

```
amqmfsck /shared/qmdata
```

Figure 40. On IBM WebSphere MQ server 1

3. Check concurrently writing to the same directory from both IBM WebSphere MQ servers.

   On both IBM WebSphere MQ servers, run the file system checker at the same time with the `-c` option.

```
amqmfsck -c /shared/qmdata
```

Figure 41. On IBM WebSphere MQ server 1

```
amqmfsck -c /shared/qmdata
```

Figure 42. On IBM WebSphere MQ server 2

4. Check waiting for and releasing locks on both IBM WebSphere MQ servers.

   On both IBM WebSphere MQ servers run the file system checker at the same time with the `-w` option.

```
amqmfsck -w /shared/qmdata
```

Figure 43. On IBM WebSphere MQ server 1

```
amqmfsck -w /shared/qmdata
```

Figure 44. On IBM WebSphere MQ server 2

5. Check for data integrity.
   a. Format the test file.

      Create a large file in the directory being tested. The file is formatted so that the subsequent phases can complete successfully. The file must be large enough that there is sufficient time to interrupt the second phase to simulate the failover. Try the default value of 262144 pages (1 GB). The program automatically reduces this default on slow file systems so that formatting completes in about 60 seconds

```
amqmfsck -f /shared/qmdata
```

The server responds with the following messages:

```
Formatting test file for data integrity test.
Test file formatted with 262144 pages of data.
```

Figure 45. On IBM WebSphere MQ server 1

   b. Write data into the test file using the file system checker while causing a failure.

      Run the test program on two servers at the same time. Start the test program on the server which is going to experience the failure, then start the test program on the server that is going to survive the failure. Cause the failure you are investigating.

      The first test program stops with an error message. The second test program obtains the lock on the test file and writes data into the test file starting where the first test program left off. Let the second test program run to completion.

Table 12. Running the data integrity check on two servers at the same time

IBM WebSphere MQ server 1	IBM WebSphere MQ server 2
amqmfsck -a /shared/qmdata	
Please start this program on a second machine with the same parameters. File lock acquired. Start a second copy of this program with the same parameters on another server. Writing data into test file. To increase the effectiveness of the test, interrupt the writing by ending the process, temporarily breaking the network connection to the networked storage, rebooting the server of turning off the power. To increase the effectiveness of the test, interrupt the writing by ending the process, temporarily breaking the network connection to the networked storage, rebooting the server of turning off the power.	amqmfsck -a /shared/qmdata
	Waiting for lock... Waiting for lock... Waiting for lock... Waiting for lock... Waiting for lock... Waiting for lock...
Turn the power off here	

Table 12. Running the data integrity check on two servers at the same time (continued)

IBM WebSphere MQ server 1	IBM WebSphere MQ server 2

	`File lock acquired.`
	`Reading test file`
	`Checking the integrity of the data read.`
	`Appending data into the test file`
	`after data already found.`
	`The test file is full of data.`
	`It is ready to be inspected for data integrity.`

The timing of the test depends on the behavior of the file system. For example, it typically takes 30 - 90 seconds for a file system to release the file locks obtained by the first program following a power outage. If you have too little time to introduce the failure before the first test program has filled the file, use the `-x` option of **`amqmfsck`** to delete the test file. Try the test from the start with a larger test file.

c. Verify the integrity of the data in the test file.

```
amqmfsck -i /shared/qmdata
```

The server responds with the following messages:

```
File lock acquired
Reading test file checking the integrity of the data read.
The data read was consistent.
The tests on the directory completed successfully.
```

Figure 46. On IBM WebSphere MQ server 2

6. Delete the test files.

```
amqmfsck -x /shared/qmdata
Test files deleted.
```

Figure 47. On IBM WebSphere MQ server 2

The server responds with the message:

```
Test files deleted.
```

Results

The program returns an exit code of zero if the tests complete successfully, and non-zero otherwise.

Examples

The first set of three examples shows the command producing minimal output.

Successful test of basic file locking on one server

```
> amqmfsck /shared/qmdata
The tests on the directory completed successfully.
```

Failed test of basic file locking on one server

```
> amqmfsck /shared/qmdata
AMQ6245: Error Calling 'write()[2]' on file '/shared/qmdata/amqmfsck.lck' error '2'.
```

Successful test of locking on two servers

Table 13. Successful locking on two servers

IBM WebSphere MQ server 1	IBM WebSphere MQ server 2

`> amqmfsck -w /shared/qmdata`
`Please start this program on a second`
`machine with the same parameters.`
`Lock acquired.`
`Press Return`
`or terminate the program to release the lock.`	

	`> amqmfsck -w /shared/qmdata`
	`Waiting for lock...`

`[ Return pressed ]`
`Lock released.`	

	`Lock acquired.`
	`The tests on the directory completed successfully`

The second set of three examples shows the same commands using verbose mode.

Successful test of basic file locking on one server

```
> amqmfsck -v /shared/qmdata
System call: stat("/shared/qmdata")'
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fchmod(fd, 0666)
System call: fstat(fd)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: write(fd)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: close(fd)
System call: fd1 = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd1, F_SETLK, F_RDLCK)
System call: fd2 = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd2, F_SETLK, F_RDLCK)
System call: close(fd2)
System call: write(fd1)
System call: close(fd1)
The tests on the directory completed successfully.
```

Failed test of basic file locking on one server

```
> amqmfsck -v /shared/qmdata
System call: stat("/shared/qmdata")
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fchmod(fd, 0666)
System call: fstat(fd)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: write(fd)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_RDLCK)
System call: fdSameFile = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fdSameFile, F_SETLK, F_RDLCK)
System call: close(fdSameFile)
System call: write(fd)
AMQxxxx: Error calling 'write()[2]' on file '/shared/qmdata/amqmfsck.lck', errno 2
(Permission denied).
```

Successful test of locking on two servers

Table 14. Successful locking on two servers - verbose mode

<table>
<tr><th>IBM WebSphere MQ server 1</th><th>IBM WebSphere MQ server 2</th></tr>
<tr><td>> amqmfsck -wv /shared/qmdata
Calling 'stat("/shared/qmdata")'
Calling 'fd = open("/shared/qmdata/amqmfsck.lkw",
O_EXCL | O_CREAT | O_RDWR, 0666)'
Calling 'fchmod(fd, 0666)'
Calling 'fstat(fd)'
Please start this program on a second
machine with the same parameters.
Calling 'fcntl(fd, F_SETLK, F_WRLCK)'
Lock acquired.
Press Return
or terminate the program to release the lock.</td><td></td></tr>
<tr><td></td><td>> amqmfsck -wv /shared/qmdata
Calling 'stat("/shared/qmdata")'
Calling 'fd = open("/shared/qmdata/amqmfsck.lkw",
O_EXCL | O_CREAT | O_RDWR,0666)'
Calling 'fd = open("/shared/qmdata/amqmfsck.lkw,
O_RDWR, 0666)'
Calling 'fcntl(fd, F_SETLK, F_WRLCK)
'Waiting for lock...</td></tr>
<tr><td>[Return pressed]
Calling 'close(fd)'
Lock released.</td><td></td></tr>
<tr><td></td><td>Calling 'fcntl(fd, F_SETLK, F_WRLCK)'
Lock acquired.
The tests on the directory completed successfully</td></tr>
</table>

Related information:

amqmfsck (file system check)

Running **amqsfhac** *to test message integrity:*

amqsfhac checks that a queue manager using networked storage maintains data integrity following a failure.

Before you begin

You require four servers for this test. Two servers for the multi-instance queue manager, one for the file system, and one for running **amqsfhac** as a IBM WebSphere MQ MQI client application.

Follow step 1 on page 122 in Procedure to set up the file system for a multi-instance queue manager.

About this task

Procedure

1. Create a multi-instance queue manager on another server, QM1, using the file system you created in step 1 on page 122 in Procedure.

   See Create a multi-instance queue manager.

2. Start the queue manager on both servers making it highly available.

   On server 1:

   ```
   strmqm -x QM1
   ```

   On server 2:

   ```
   strmqm -x QM1
   ```

3. Set up the client connection to run **amqsfhac**.

a. Use the procedure in Verifying a client installation to set up a client connection, or the example scripts in Reconnectable client samples.

b. Modify the client channel to have two IP addresses, corresponding to the two servers running QM1.

   In the example script, modify:

   ```
   DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN)  TRPTYPE(TCP) +
   CONNAME('LOCALHOST(2345)') QMNAME(QM1) REPLACE
   ```

   To:

   ```
   DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN)  TRPTYPE(TCP) +
   CONNAME('server1(2345),server2(2345)') QMNAME(QM1) REPLACE
   ```

   Where server1 and server2 are the host names of the two servers, and 2345 is the port that the channel listener is listening on. Usually this defaults to 1414. You can use 1414 with the default listener configuration.

4. Create two local queues on QM1 for the test. Run the following MQSC script:

   ```
   DEFINE QLOCAL(TARGETQ) REPLACE
   DEFINE QLOCAL(SIDEQ) REPLACE
   ```

5. Test the configuration with **amqsfhac**

   ```
   amqsfhac QM1 TARGETQ SIDEQ 2 2 2
   ```

6. Test message integrity while you are testing file system integrity.

   Run **amqsfhac** during step 5 on page 123 of Procedure.

   ```
   amqsfhac QM1 TARGETQ SIDEQ 10 20 0
   ```

   If you stop the active queue manager instance, **amqsfhac** reconnects to the other queue manager instance once it has become active. Restart the stopped queue manager instance again, so that you can reverse the failure in your next test. You will probably need to increase the number of iterations based on experimentation with your environment so that the test program runs for sufficient time for the failover to occur.

Results

An example of running **amqsfhac** in step 6 is shown in Figure 48 on page 128. The test is a success.

If the test detected a problem, the output would report the failure. In some test runs MQRC_CALL_INTERRUPTED might report “Resolving to backed out”. It makes no difference to the result. The outcome depends on whether the write to disk was committed by the networked file storage before or after the failure took place.

```
Sample AMQSFHAC start
qmname = QM1
qname = TARGETQ
sidename = SIDEQ
transize = 10
iterations = 20
verbose = 0
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Resolving MQRC_CALL_INTERRUPTED
MQGET browse side tranid=14 pSideinfo->tranid=14
Resolving to committed
Iteration 7
Iteration 8
Iteration 9
Iteration 10
Iteration 11
Iteration 12
Iteration 13
Iteration 14
Iteration 15
Iteration 16
Iteration 17
Iteration 18
Iteration 19
Sample AMQSFHAC end
```

Figure 48. Output from a successful run of **amqsfhac**

Related information:

High availability sample programs

Sharing WebSphere MQ files

Some WebSphere MQ files are accessed exclusively by an active queue manager, other files are shared.

WebSphere MQ files are split into program files and data files. Program files are typically installed locally on each server running WebSphere MQ. Queue managers share access to data files and directories in the default data directory. They require exclusive access to their own queue manager directory trees contained in each of the `qmgrs` and `log` directories shown in Figure 49 on page 129.

Figure 49 on page 129 is a high-level view of the WebSphere MQ directory structure. It shows the directories which can be shared between queue managers and made remote. The details vary by platform. The dotted lines indicate configurable paths.

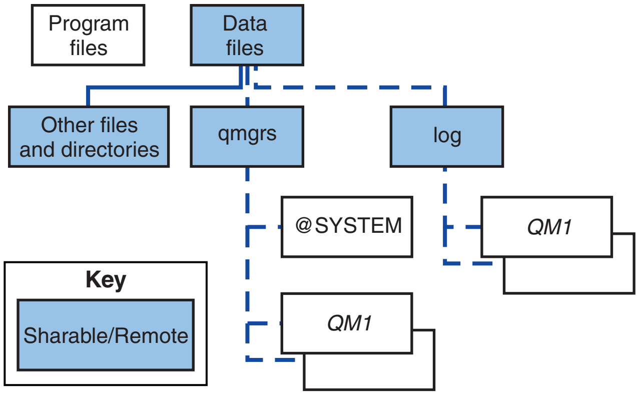


Figure 49. Overall view of WebSphere MQ directory structure

Program files

The program files directory is typically left in the default location, is local, and shared by all the queue managers on the server.

Data files

The data files directory is typically local in the default location, `/var/mqm` on UNIX and Linux systems and configurable on installation on Windows. It is shared between queue managers. You can make the default location remote, but do not share it between different installations of WebSphere MQ. The `DefaultPrefix` attribute in the WebSphere MQ configuration points to this path.

qmgrs From v7.0.1, there are two alternative ways to specify the location of queue manager data.

Using Prefix

The `Prefix` attribute specifies the location of the `qmgrs` directory. WebSphere MQ constructs the queue manager directory name from the queue manager name and creates it as a subdirectory of the `qmgrs` directory.

The `Prefix` attribute is located in the QueueManager stanza, and is inherited from the value in the `DefaultPrefix` attribute. By default, for administrative simplicity, queue managers typically share the same `qmgrs` directory.

The `QueueManager` stanza is in the `mqs.ini` file.

If you change the location of the `qmgrs` directory for any queue manager, you must change the value of its `Prefix` attribute.

The `Prefix` attribute for the `QM1` directory in Figure 49 for a UNIX and Linux platform is,

```
Prefix=/var/mqm
```

Using DataPath

The `DataPath` attribute specifies the location of the queue manager data directory.

The `DataPath` attribute specifies the complete path, including the name of the queue manager data directory. The `DataPath` attribute is unlike the `Prefix` attribute, which specifies an incomplete path to the queue manager data directory.

The `DataPath` attribute, if it is specified, is located in the QueueManager stanza. If it has been specified, it takes precedence over any value in the `Prefix` attribute.

The `QueueManager` stanza is in the `mqs.ini` file.

If you change the location of the queue manager data directory for any queue manager you must change the value of the `DataPath` attribute.

The `DataPath` attribute for the `QM1` directory in Figure 49 on page 129 for a UNIX or Linux platform is,

```
DataPath=/var/mqm/qmgrs/QM1
```

`log`

The log directory is specified separately for each queue manager in the `Log` stanza in the queue manager configuration. The queue manager configuration is in `qm.ini`.

***DataPath*/*QmgrName*/@IPCC subdirectories**

The *DataPath*/*QmgrName*/`@IPCC` subdirectories are in the shared directory path. They are used to construct the directory path for IPC file system objects. They need to distinguish the namespace of a queue manager when a queue manager is shared between systems. Before V7.0.1, a queue manager was only used on one system. One set of subdirectories was sufficient to define the directory path to IPC file system objects, see Figure 50.

```
DataPath/QmgrName/@IPCC/esem
```

Figure 50. Example IPC subdirectory, pre-V7.0.1

In V7.0.1, and later, the IPC file system objects have to be distinguished by system. A subdirectory, for each system the queue manager runs on, is added to the directory path, see Figure 51.

```
DataPath/QmgrName/@IPCC/esem/myHostName/
```

Figure 51. Example IPC subdirectory, V7.0.1 and subsequent releases

myHostName is up to the first 20 characters of the host name returned by the operating system. On some systems, the host name might be up to 64 characters in length before truncation. The generated value of *myHostName* might cause a problem for two reasons:

1. The first 20 characters are not unique.
2. The host name is generated by a DHCP algorithm that does not always allocate the same host name to a system.

In these cases, set *myHostName* using the environment variable, `MQC_IPC_HOST`; see Figure 52.

```
export MQS_IPC_HOST=myHostName
```

Figure 52. Example: setting `MQC_IPC_HOST`

Other files and directories

Other files and directories, such as the directory containing trace files, and the common error log, are normally shared and kept on the local file system.

Up until v7.0.1, WebSphere MQ relied upon external management to guarantee queue managers exclusive access to the queue manager data and log files. From v7.0.1 onwards, with support of shared file systems, WebSphere MQ manages exclusive access to these files using file system locks. A file system lock allows only one instance of a particular queue manager to be active at a time.

When you start the first instance of a particular queue manager it takes ownership of its queue manager directory. If you start a second instance, it can only take ownership if the first instance has stopped. If the first queue manager is still running, the second instance fails to start, and reports the queue manager is running elsewhere. If the first queue manager has stopped, then the second queue manager takes over ownership of the queue manager files and becomes the running queue manager.

You can automate the procedure of the second queue manager taking over from the first. Start the first queue manager with the `strmqm -x` option that permits another queue manager to take over from it. The second queue manager then waits until the queue manager files are unlocked before attempting to take over ownership of the queue manager files, and start.

Directory structure on UNIX and Linux systems

The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Use the flexible directory structure of WebSphere MQ to take advantage of shared file systems for running multi-instance queue managers.

Use the command **crtmqm** *QM1* to create the directory structure shown in Figure 53 where R is the release of the product. It is a typical directory structure for a queue manager created on a WebSphere MQ system from v7.0.1 onwards. Some directories, files and .ini attribute settings are omitted for clarity, and another queue manager name might be altered by mangling. The names of the file systems vary on different systems.

In a typical installation, every queue manager that you create points to common log and qmgrs directories on the local file system. In a multi-instance configuration, the log and qmgrs directories are on a network file system shared with another installation of WebSphere MQ.

Figure 53 shows the default configuration for WebSphere MQ v7.R on AIX where R is the release of the product. For examples of alternative multi-instance configurations, see "Example directory configurations on UNIX and Linux systems" on page 135.

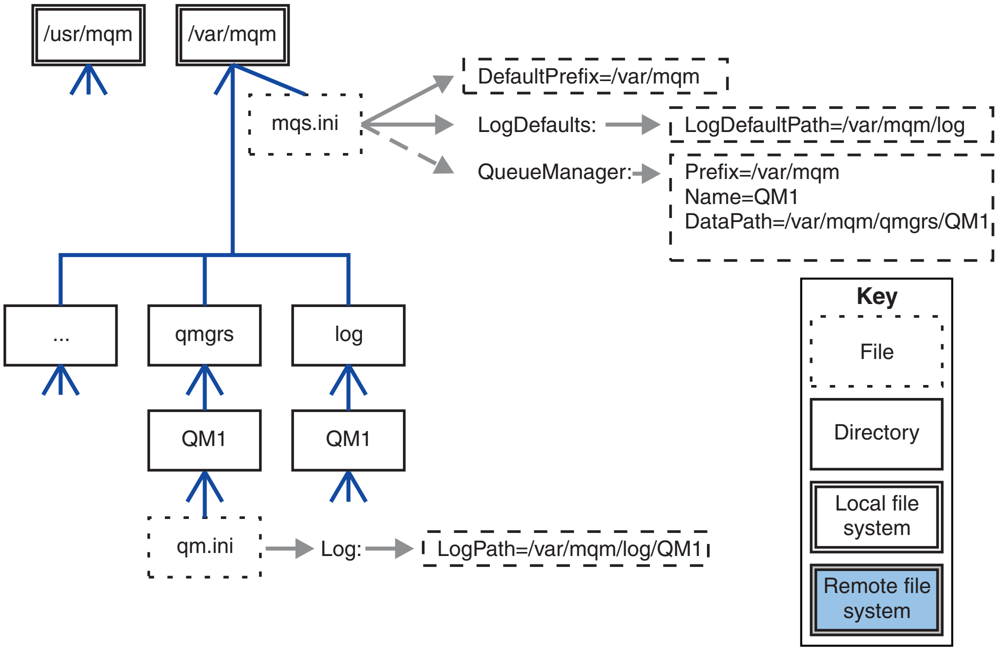


Figure 53. Example default WebSphere MQ v7.R directory structure for UNIX and Linux systems

The product is installed into /usr/mqm on AIX and /opt/mqm on the other systems, by default. The working directories are installed into the /var/mqm directory.

Note: If you created the /var/mqm file system prior to installing IBM WebSphere MQ, ensure that the mqm user has full directory permissions, for example, file mode 755.

The log and qmgrs directories are shown in their default locations as defined by the default values of the LogDefaultPath and DefaultPrefix attributes in the mqs.ini file. When a queue manager is created, by default the queue manager data directory is created in *DefaultPrefix*/qmgrs , and the log file directory in *LogDefaultPath* /log. LogDefaultPath and DefaultPrefix only effects where queue managers and log files are created by default. The actual location of a queue manager directory is saved in the mqs.ini file, and the location of the log file directory is saved in the qm.ini file.

The log file directory for a queue manager is defined in the qm.ini file in the LogPath attribute. Use the -ld option on the **crtmqm** command to set the LogPath attribute for a queue manager; for example, **crtmqm** -ld *LogPath* QM1. If you omit the ld parameter the value of LogDefaultPath is used instead.

The queue manager data directory is defined in the DataPath attribute in the QueueManager stanza in the mqs.ini file. Use the -md option on the **crtmqm** command to set the DataPath for a queue manager; for example, **crtmqm -** md *DataPath* QM1 . If you omit the md parameter the value of the DefaultPrefix or Prefix attribute is used instead. Prefix takes precedence over DefaultPrefix .

Typically, create QM1 specifying both the log and data directories in a single command.

```
crtmqm
 -md DataPath -ld
 LogPath QM1
```

You can modify the location of a queue manager log and data directories of an existing queue manager by editing the DataPath and LogPath attributes in the qm.ini file when the queue manager is stopped.

The path to the errors directory, like the paths to all the other directories in /var/mqm , is not modifiable. However the directories can be mounted on different file systems, or symbolically linked to different directories.

Directory content on UNIX and Linux systems:

Content of the directories associated with a queue manager.

For information about the location of the product files, see Choosing an installation location

For information about alternative directory configurations, see “Planning file system support” on page 119.

In Figure 54 on page 133, the layout is representative of WebSphere MQ after a queue manager has been in use for some time. The actual structure that you have depends on which operations have occurred on the queue manager.

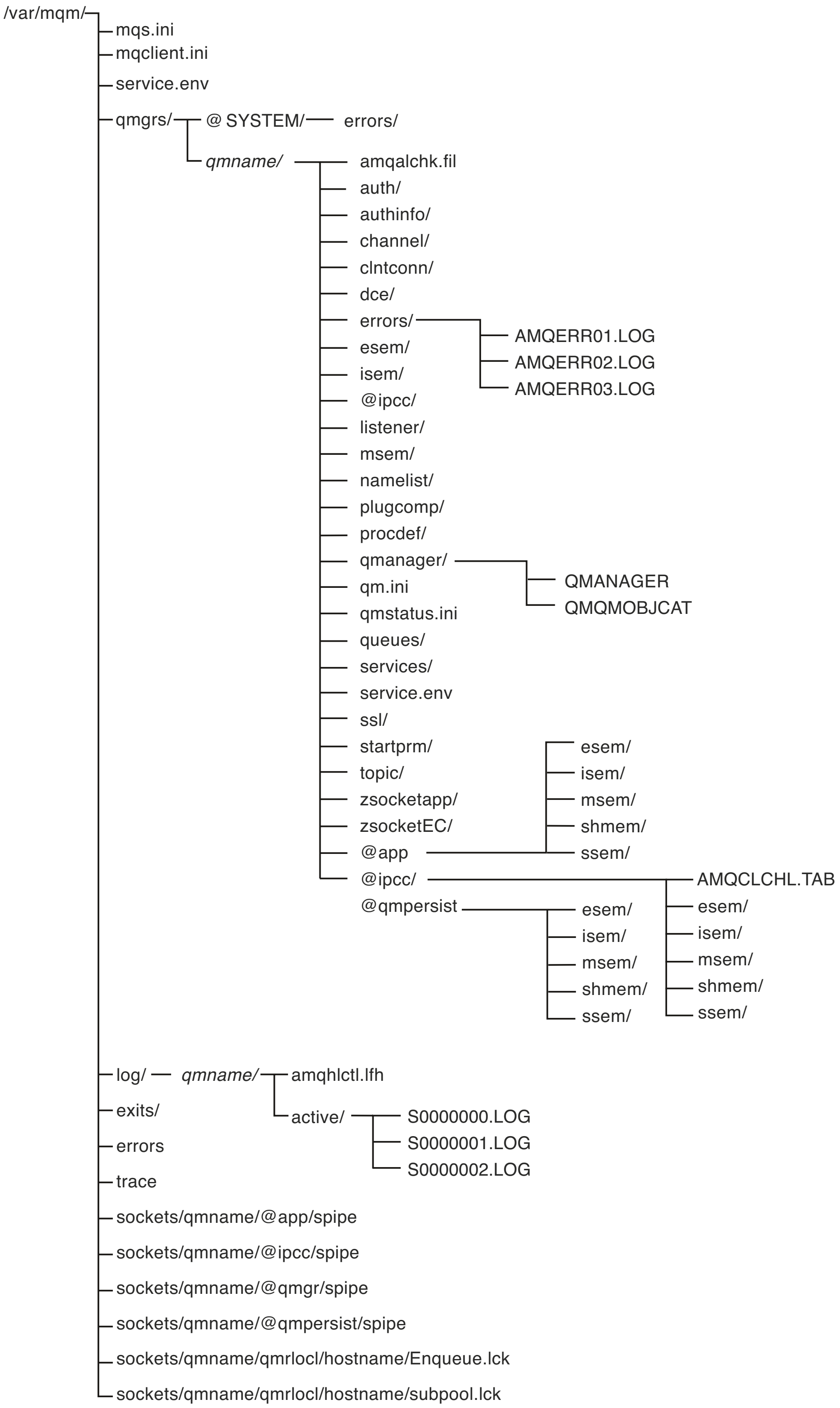


Figure 54. Default directory structure (UNIX systems) after a queue manager has been started

/var/mqm/

The /var/mqm directory contains configuration files and output directories that apply to a WebSphere MQ installation as a whole, and not to an individual queue manager.

Table 15. Documented content of the /var/mqm directory on UNIX systems

mqs.ini	WebSphere MQ installation wide configuration file, read when a queue manager starts. File path modifiable using the **AMQ_MQS_INI_LOCATION** environment variable. Ensure this is set and exported in the shell in which the **strmqm** command is run.
mqclient.ini	Default client configuration file read by WebSphere MQ MQI client programs. File path modifiable using the **MQCLNTCF** environment variable.
service.env	Contains machine scope environment variables for a service process. File path fixed.
errors/	Machine scope error logs, and FFST files. Directory path fixed. See also, FFST™: WebSphere MQ for UNIX and Linux systems.
sockets/	Contains information for each queue manager for system use only.
trace/	Trace files. Directory path fixed.
exits/	Default directory containing user channel exit programs. Location modifiable in ApiExit stanzas in the mqs.ini file.
exits64/	

/var/mqm/qmgrs/*qmname*/

/var/mqm/qmgrs/*qmname*/ contains directories and files for a queue manager. The directory is locked for exclusive access by the active queue manager instance. The directory path is directly modifiable in the mqs.ini file, or by using the **md** option of the **crtmqm** command.

Table 16. Documented contents of the /var/mqm/qmgrs/qmname directory on UNIX systems

qm.ini	Queue manager configuration file, read when a queue manager starts.
errors/	Queue manager scope error logs. *qmname* = @system contains channel-related messages for an unknown or unavailable queue manager.
@ipcc/ AMQCLCHL.TAB	Default client channel control table, created by the WebSphere MQ server, and read by WebSphere MQ MQI client programs. File path modifiable using the **MQCHLLIB** and **MQCHLTAB** environment variables.
qmanager	Queue manager object file: QMANAGER Queue manager object catalog: QMQMOBJCAT

Table 16. Documented contents of the `/var/mqm/qmgrs/qmname` directory on UNIX systems (continued)

authinfo/	Each object defined within the queue manager is associated with a file in these directories. The file name approximately matches the definition name; see, Understanding WebSphere MQ file names .
channel/	
clntconn/	
listener/	
namelist/	
procdef/	
queues/	
services/	
topics/	
...	Other directories used by WebSphere MQ, such as @ipcc, to be modified only by WebSphere MQ.

/var/mqm/log/*qmname*/

`/var/mqm/log/`*`qmname`*`/` contains the queue manager log files. The directory is locked for exclusive access by the active queue manager instance. The directory path is modifiable in the `qm.ini` file, or by using the **ld** option of the **crtmqm** command.

Table 17. Documented contents of the `/var/mqm/log/qmname` directory on UNIX systems

amqhlctl.lfh	Log control file.
active/	This directory contains the log files numbered S0000000.LOG, S0000001.LOG, S0000002.LOG, and so on.

Example directory configurations on UNIX and Linux systems:

Examples of alternative file system configurations on UNIX and Linux systems.

You can customize the WebSphere MQ directory structure in various ways to achieve a number of different objectives.

- Place the `qmgrs` and `log` directories on remote shared file systems to configure a multi-instance queue manager.
- Use separate file systems for the data and log directories, and allocate the directories to different disks, to improve performance by reducing I/O contention.
- Use faster storage devices for directories that have a greater effect on performance. Physical device latency is frequently a more important factor in the performance of persistent messaging than whether a device is mounted locally or remotely. The following list shows which directories are most and least performance sensitive.
  1. `log`
  2. `qmgrs`
  3. Other directories, including `/usr/mqm`
- Create the `qmgrs` and `log` directories on file systems that are allocated to storage with good resilience, such as a redundant disk array, for example.
- It is better to store the common error logs in `var/mqm/errors`, locally, rather than on a network file system, so that error relating to the network file system can be logged.

Figure 55 on page 136 is a template from which alternative WebSphere MQ directory structures are derived. In the template, dotted lines represent paths that are configurable. In the examples, the dotted lines are replaced by solid lines that correspond to the configuration information stored in the AMQ_MQS_INI_LOCATION environment variable, and in the mqs.ini and qm.ini files.

Note: The path information is shown as it appears in the mqs.ini or qm.ini files. If you supply path parameters in the **crtmqm** command, omit the name of the queue manager directory: the queue manager name is added to the path by WebSphere MQ after it has been mangled.

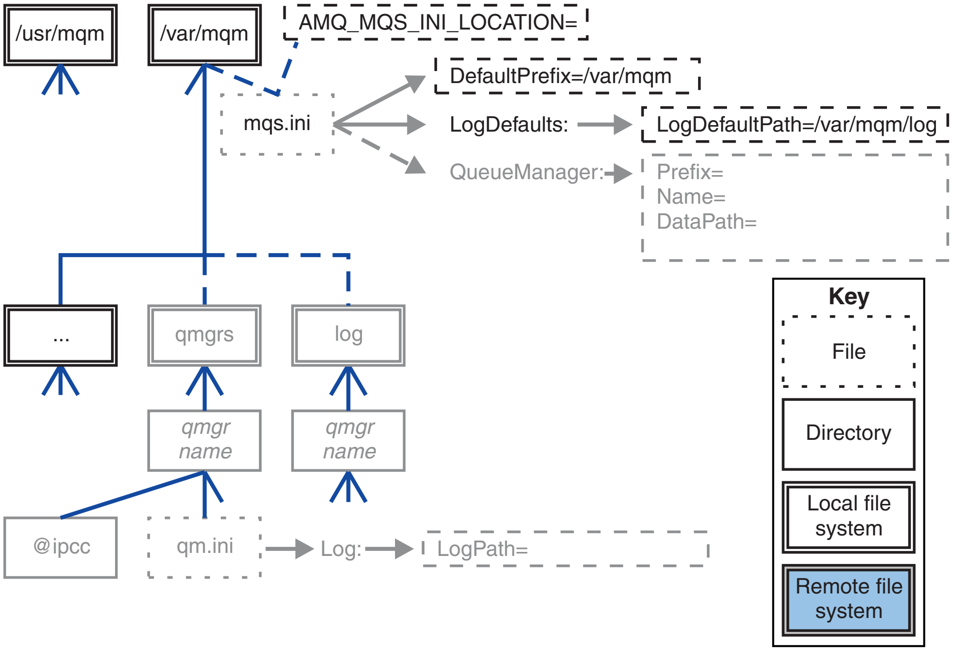


Figure 55. Directory structure pattern template

Examples of configured directory structures follow. The first example shows a typical default directory structure for WebSphere MQ v7.0.1 created by issuing the **crtmqm** *QM1* command. The second example shows how a typical directory structure appears for a queue manager created using a WebSphere MQ release earlier than v7.0.1. The directory structure does not change.

Queue managers newly created in version 7.0.1 have a different configuration file to earlier releases of v7. If you need to remove the v7.0.1 fix pack to revert to v7.0.0.2, you need to re-create the configuration files. You might only need to use the Prefix attribute to define the path to the new queue manager data directory, or you might need to move the queue manager data directory and log directories to a different location. The safest way to reconfigure the queue manager is to save the queue manager data and log directories, delete and re-create the queue manager, and then replace the data and log directories in their new location, with the ones that have been saved.

Typical directory structure for release v7.0.1 onwards

Figure 56 on page 137 is the default directory structure created in v7.0.1 by issuing the command **crtmqm** *QM1*.

The mqs.ini file has a stanza for the QM1 queue manager, created by referring to the value of DefaultPrefix . The Log stanza in the qm.ini file has a value for LogPath, set by reference to LogDefaultPath in mqs.ini .

Use the optional **crtmqm** parameters to override the default values of DataPath and LogPath.

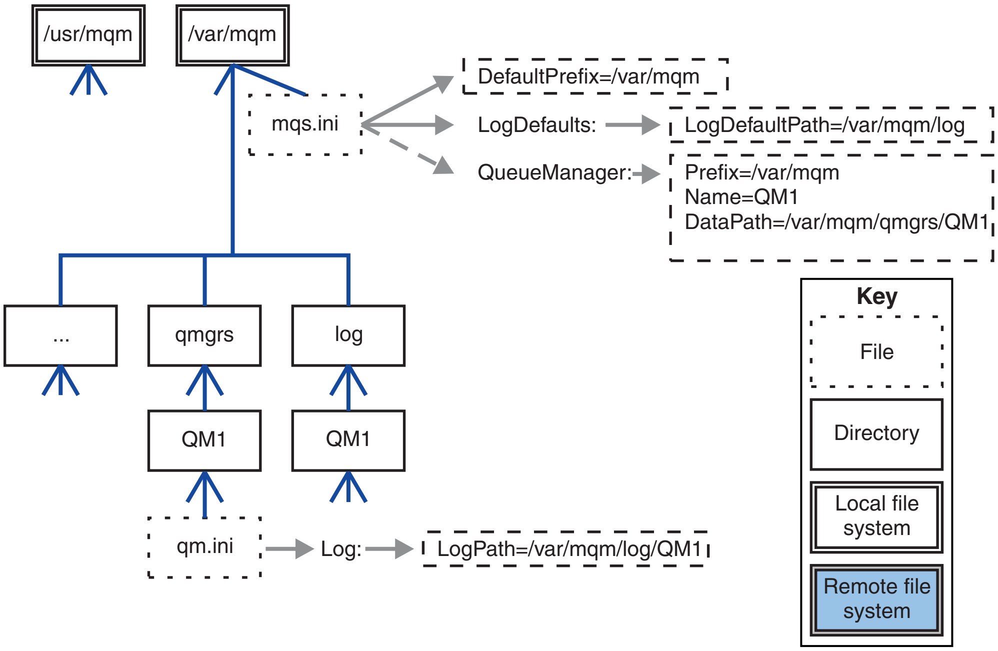


Figure 56. Example default WebSphere MQ v7.R directory structure for UNIX and Linux systems

Typical directory structure for releases earlier than v7.0.1

The `DataPath` attribute did not exist before WebSphere MQ v7.0.1; the attribute is not present in the `mqs.ini` file. The location of the `qmgrs` directory was configured using the `Prefix` attribute. The location of individual directories could be configured by using symbolic links to point to different file system locations.

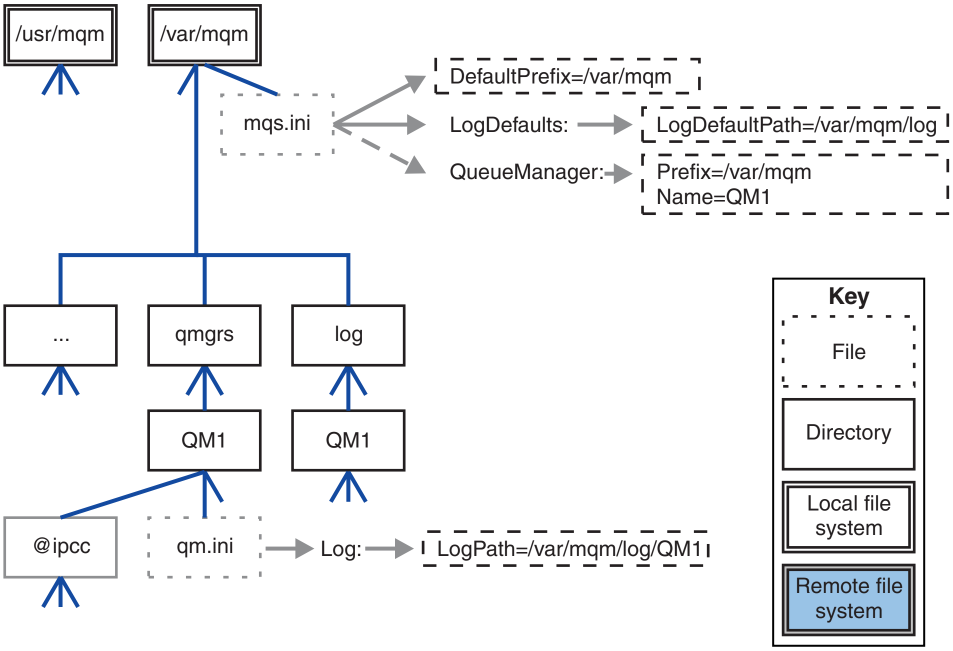


Figure 57. Typical directory structure for releases earlier than v7.0.1

Share default `qmgrs` and `log` directories (Release v7.0.1 onwards)

An alternative to “Share everything (Release v7.0.1 onwards)” on page 139, is to share the `qmgrs` and `log` directories separately (Figure 58). In this configuration, there is no need to set `AMQ_MQS_INI_LOCATION` as the default mqs.ini is stored in the local `/var/mqm` file system. The files and directories, such as `mqclient.ini` and `mqserver.ini` are also not shared.

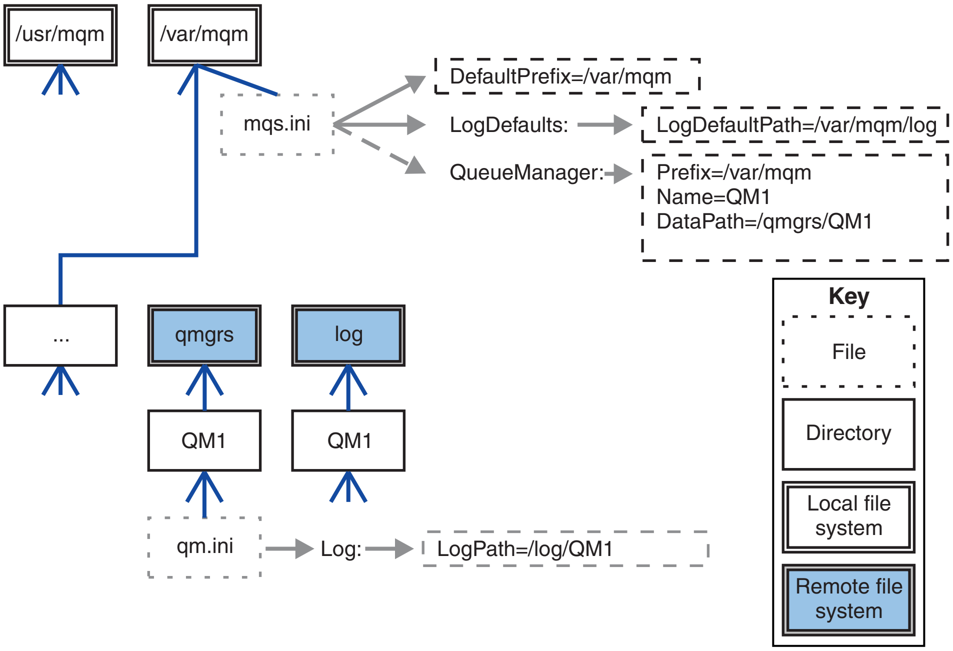


Figure 58. Share `qmgrs` and `log` directories

Share named qmgrs and log directories (Release v7.0.1 onwards)

The configuration in Figure 59 places the log and qmgrs in a common named remote shared file system called /ha . The same physical configuration can be created in two different ways.

1. Set LogDefaultPath=/ha and then run the command, **crtmqm** -md */ha/qmgrs* QM1. The result is exactly as illustrated in Figure 59.
2. Leave the default paths unchanged and then run the command, **crtmqm** -ld */ha/log* -md */ha/qmgrs* QM1.

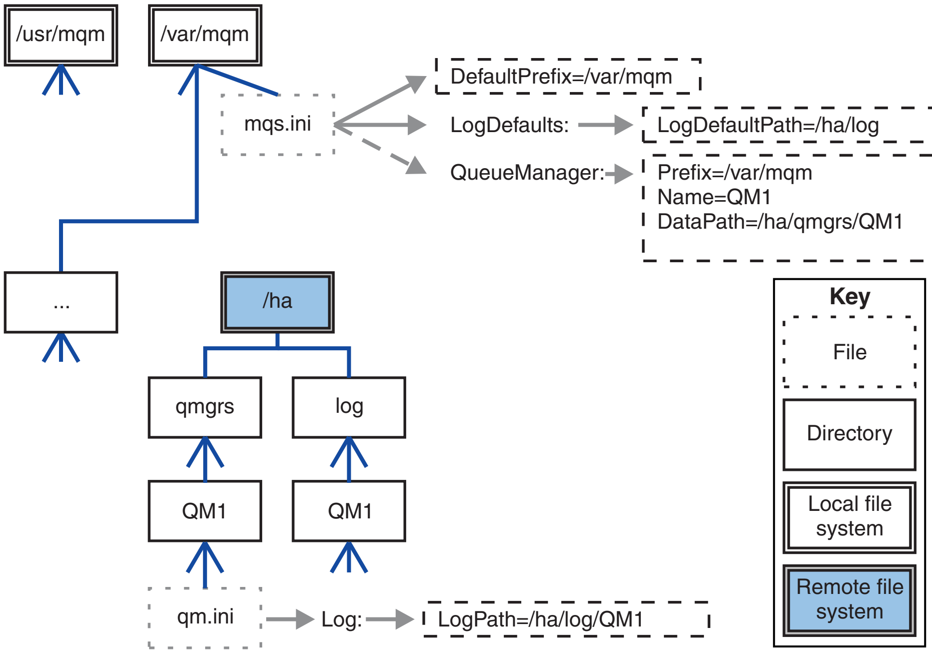


Figure 59. Share named qmgrs and log directories

Share everything (Release v7.0.1 onwards)

Figure 60 on page 140 is a simple configuration for system with fast networked file storage.

Mount /var/mqm as a remote shared file system. By default, when you start QM1, it looks for /var/mqm, finds it on the shared file system, and reads the mqs.ini file in /var/mqm. Rather than use the single /var/mqm/mqs.ini file for queue managers on all your servers, you can set the AMQ_MQS_INI_LOCATION environment variable on each server to point to different mqs.ini files.

Note: The contents of the generic error file in /var/mqm/errors/ are shared between queue managers on different servers.

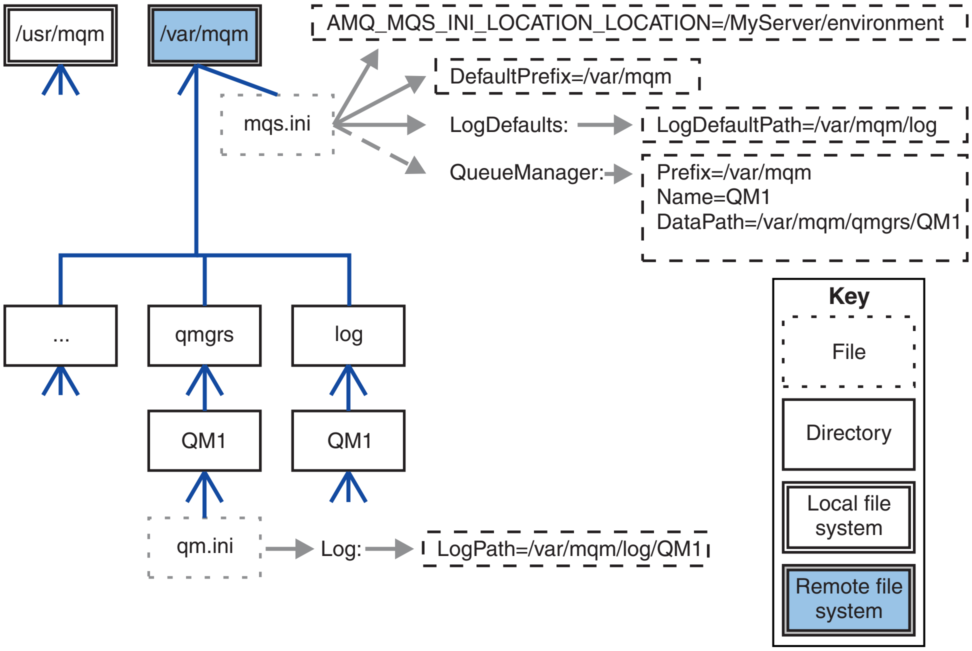


Figure 60. Share everything

Note that you cannot use this for multi-instance queue managers. The reason is that it is necessary for each host in a multi-instance queue manager to have its own local copy of `/var/mqm` to keep track of local data, such as semaphores and shared memory. These entities cannot be shared across hosts.

Directory structure on Windows systems

How to find queue manager configuration information and directories on Windows.

The default directory for IBM WebSphere MQ for Windows installation is:

32 bit `C:\Program Files\IBM\WebSphere MQ`

64 bit `C:\Program Files (x86)\IBM\WebSphere MQ`

The installation information is stored in the Windows registry. The registry key where IBM WebSphere MQ information is stored is:

32 bit `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere MQ\`

64 bit `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\IBM\WebSphere MQ\`

Each installation has a specific sub key:

```
Installation\<InstallationName>\
```

The path that points to the IBM WebSphere MQ data directory is stored in a string value named `WorkPath` and the default directory for logs is stored in `LogDefaultPath`. Queue manager data directories are created in *`WorkPath`*`\qmgrs\`*`Qmgrname`*. Queue manager logs are created in *`LogDefaultPath`*`\`*`QmgrName`*. See Figure 61 on page 141.

If you define the queue manager data and log directories when installing IBM WebSphere MQ, then *WorkPath* and *LogDefaultPath* are updated with the customized path information.

WorkPath and *LogDefaultPath* are only used to create a queue manager.

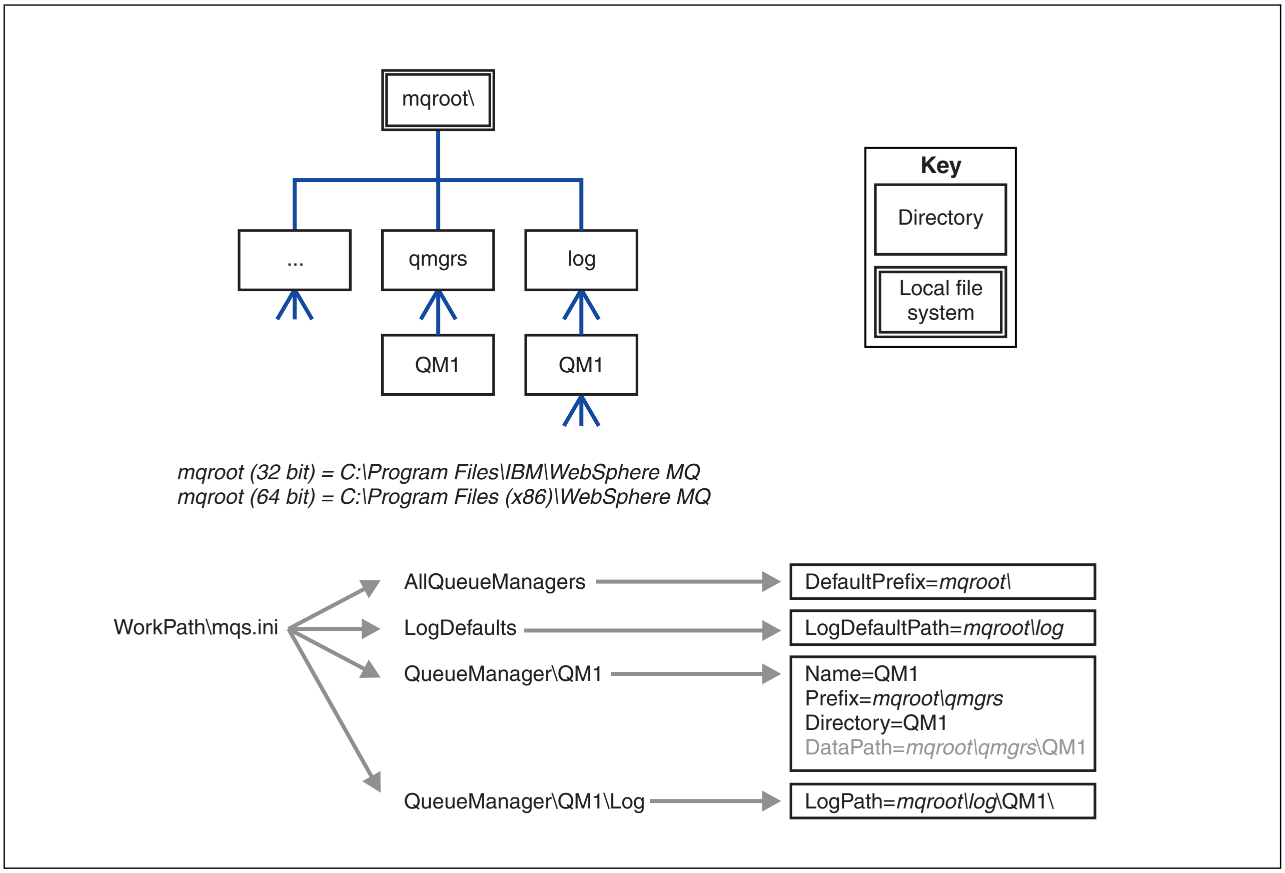


Figure 61. Default directory structure for Windows

Multi-instance queue managers

To configure a multi-instance queue manager, the log and data directories must be placed on networked storage, preferably on a different server to any of the servers that are running instances of the queue manager.

Two parameters are provided on the **crtmqm** command, **-md** and **-ld**, to make it easier specify the location of the queue manager data and log directories. The effect of specifying the **-md** parameter is fourfold:

1. The `mqs.ini` stanza `QueueManager\`*`QmgrName`* contains a new variable, *DataPath*, which points to the queue manager data directory. Unlike the *Prefix* variable, the path includes the name of the queue manager directory.
2. The queue manager configuration information stored in the mqs.ini file is reduced to *Name*, *Prefix*, *Directory* and *DataPath*.

Directory content:

Lists the location and content of WebSphere MQ directories.

A WebSphere MQ configuration has three main sets of files and directories,

1. Executable, and other read-only files such as the readme file, the WebSphere MQ Explorer plug-in and help files, and licenses files, that are only updated when maintenance is applied. These files are described in Table 18 on page 142.
2. Potentially modifiable files and directories that are not specific to a particular queue manager. These files and directories are described in Table 19 on page 142.

3. Files and directories that are specific to each queue manager on a server. These files and directories are described in Table 18

Resource directories and files

The resource directories and files contain all the executable code and resources to run a queue manager. The variable, *FilePath*, in the installation specific IBM WebSphere MQ configuration registry key, contains the path to the resource directories.

Table 18. Directories and files in the `FilePath` directory

File path	Contents
FilePath\bin	Commands and DLLs
FilePath\bin64	Commands and DLLs (64 bit)
FilePath\conv	Data conversion tables
FilePath\doc	Wizard help files
FilePath\MQExplorer	Explorer and Explorer help Eclipse plug-ins
FilePath\gskit8	Global security kit
FilePath\java	Java resources, including JRE
FilePath\licenses	License information
FilePath\Non_IBM_License	License information
FilePath\properties	Used internally
FilePath\Tivoli	
FilePath\tools	Development resources and samples
FilePath\Uninst	Used internally
FilePath\README.TXT	Readme file

Directories not specific to a queue manager

Some directories contain files, such as trace files and error logs, that are not specific to a specific queue manager. The *DefaultPrefix* variable contains the path to these directories. *DefaultPrefix* is part of the `AllQueueManagers` stanza.

Table 19. Directories and files in `DefaultPrefix` directory

File path	Contents
DefaultPrefix\Config	Used internally
DefaultPrefix\conv	ccsid.tbl data conversion control file, described in Data conversion
DefaultPrefix\errors	Non queue manager error logs, AMQERR*nn*.LOG
DefaultPrefix\exits	Channel exit programs
DefaultPrefix\exits64	Channel exit programs (64 bit)
DefaultPrefix\ipc	Not used
DefaultPrefix\Qmgrs	Described in Table 20 on page 143
DefaultPrefix\trace	Trace files
DefaultPrefix\amqmjpse.txt	Used internally

Queue manager directories

When you create a queue manager, a new set of directories, specific to the queue manager, is created.

If you create a queue manager with the **-md** *filepath* parameter, the path is stored in the *DataPath* variable in the queue manager stanza of the mqs.ini file. If you create a queue manager without setting the **-md** *filepath* parameter, the queue manager directories are created in the path stored in *DefaultPrefix*, and the path is copied into the *Prefix* variable in the queue manager stanza of the mqs.ini file.

Table 20. Directories and files in DataPath and Prefix/Qmgrs/QmgrName directories

File path	Contents
DataPath\@ipcc	Default location for AMQCLCHL.TAB, the client connection table.
DataPath\authinfo	Used internally.
DataPath\channel	
DataPath\clntconn	
DataPath\errors	Error logs, AMQERR*nn*.LOG
DataPath\listener	Used internally.
DataPath\namelist	
DataPath\plugcomp	
DataPath\procdef	
DataPath\qmanager	
DataPath\queues	
DataPath\services	
DataPath\ssl	
DataPath\startprm	
DataPath\topic	
DataPath\active	
DataPath\active.dat	
DataPath\amqalchk.fil	
DataPath\master	
DataPath\master.dat	
DataPath\qm.ini	Queue manager configuration
DataPath\qmstatus.ini	Queue manager status
Prefix\Qmgrs*QmgrName*	Used internally
Prefix\Qmgrs\@SYSTEM	Not used
Prefix\Qmgrs\@SYSTEM\errors	

WebSphere MQ and UNIX System V IPC resources

A queue manager uses some IPC resources. Use **ipcs -a** to find out what resources are being used.

This information applies to WebSphere MQ running on UNIX and Linux systems only.

WebSphere MQ uses System V interprocess communication (IPC) resources (*semaphores* and *shared memory segments*) to store and pass data between system components. These resources are used by queue manager processes and applications that connect to the queue manager. WebSphere MQ MQI clients do not use IPC resources, except for WebSphere MQ trace control. Use the UNIX command **ipcs -a** to get full information on the number and size of the IPC resources currently in use on the machine.

Shared memory on AIX

If certain application types fail to connect because of an AIX memory limitation, in most cases this can be resolved by setting the environment variable EXTSHM=ON.

Some 32-bit processes on AIX might encounter an operating system limitation that affects their ability to connect to WebSphere MQ queue managers. Every standard connection to WebSphere MQ uses shared memory, but unlike other UNIX and Linux platforms, AIX allows 32-bit processes to attach only 11 shared memory sets.

Most 32-bit processes will not encounter this limit, but applications with high memory requirements might fail to connect to WebSphere MQ with reason code 2102: MQRC_RESOURCE_PROBLEM. The following application types might see this error:

- Programs running in 32-bit Java virtual machines
- Programs using the large or very large memory models
- Programs connecting to many queue managers or databases
- Programs that attach to shared memory sets on their own

AIX offers an extended shared memory feature for 32-bit processes that allows them to attach more shared memory. To run an application with this feature, export the environment variable EXTSHM=ON before starting your queue managers and your program. The EXTSHM=ON feature prevents this error in most cases, but it is incompatible with programs that use the SHM_SIZE option of the shmctl function.

WebSphere MQ MQI client applications and all 64-bit processes are unaffected by this limitation. They can connect to WebSphere MQ queue managers regardless of whether EXTSHM has been set.

WebSphere MQ and UNIX Process Priority

Good practices when setting process priority *nice* values.

This information applies to WebSphere MQ running on UNIX and Linux systems only.

If you run a process in the background, that process can be given a higher *nice* value (and hence lower priority) by the invoking shell. This might have general WebSphere MQ performance implications. In highly-stressed situations, if there are many ready-to-run threads at a higher priority and some at a lower priority, operating system scheduling characteristics can deprive the lower priority threads of processor time.

It is good practice that independently started processes associated with queue managers, such as **runmqlsr**, have the same *nice* values as the queue manager they are associated with. Ensure the shell does not assign a higher *nice* value to these background processes. For example, in ksh, use the setting "`set +o bgnice`" to stop ksh from raising the *nice* value of background processes. You can verify the *nice* values of running processes by examining the *NI* column of a "ps -efl" listing.

Also, start WebSphere MQ application processes with the same *nice* value as the queue manager. If they run with different *nice* values, an application thread might block a queue manager thread, or vice versa, causing performance to degrade.

Planning your IBM WebSphere MQ client environment on HP Integrity NonStop Server

When you are planning your IBM WebSphere MQ environment, you must consider the HP Integrity NonStop Server environment, and HP NonStop TMF. Use the information to plan the environment where IBM WebSphere MQ client for HP Integrity NonStop Server runs.

Before you plan your IBM WebSphere MQ client for HP Integrity NonStop Server architecture, familiarize yourself with the basic IBM WebSphere MQ client for HP Integrity NonStop Server concepts, see the topics in IBM WebSphere MQ client for HP Integrity NonStop Server technical overview.

Preparing the HP Integrity NonStop Server environment

Before installation, the environment must be prepared depending on whether the installation is to be verified immediately or not.

For the installation, you require the following items:

- A user ID that meets the requirements. For details about user ID requirements, see "Setting up the user and group on HP Integrity NonStop Server" on page 199.
- Verified locations in the OSS and Guardian file systems that can be for the installation files.
- An operational OSS shell and OSS file system. You can verify the file system by doing the following tasks:
  - Log on to the OSS environment (shell). Ensure that you have write access to the OSS installation root directory you intend to use.
  - Log on to the TACL environment using the user ID in the MQM group. Verify that the volume you intend to use meets the requirements and is accessible to you, and that the subvolume does not exist.

  You can login to both OSS or TACL using either an alias, if you have one, or your full principal.

If you intend to proceed immediately to verify that the installation is usable, you might also need the following optional items:

- An operational and accessible Local Sockets subsystem in the OSS environment.
- An operational TCP/IP subsystem.

If you intend to use TMF coordinated global units of work, you will need the following items:

- An operational TMF subsystem.
- An operational Pathway (TS/MP) subsystem.

Work with your systems administrator if you are in any doubt about the status of these critical subsystems.

IBM WebSphere MQ and HP NonStop TMF

IBM WebSphere MQ client for HP Integrity NonStop Server can participate in HP NonStop Transaction Management Facility (HP NonStop TMF) coordinated units of work. Coordinating transactions with HP NonStop TMF is only supported where the queue manager is at IBM WebSphere MQ Version 7.1 or later.

The IBM WebSphere MQ provided TMF/Gateway converts transactions from TMF coordination into eXtended Architecture (XA) transaction coordination to communicate with the remote queue manager. The IBM WebSphere MQ provided TMF/Gateway is the bridge between TMF and queue manager transactions, using the services provided by HP NonStop TMF, and has been designed to run in a Pathway environment.

HP NonStop TMF software provides transaction protection and database consistency in demanding environments. For more information about HP NonStop TMF, see http://h20223.www2.hp.com/NonStopComputing/downloads/TMF.pdf.

For information about how to configure the IBM WebSphere MQ provided TMF/Gateway, see "Configuring HP Integrity NonStop Server" on page 1175.

Using HP NonStop TMF

The HP NonStop Transaction Management Facility (TMF) is the native transaction manager on HP Integrity NonStop Server and is integrated with the file system and the relational database managers, SQL/MP, and SQL/MX.

IBM WebSphere MQ client for HP Integrity NonStop Server can use TMF to coordinate global units of work.

To coordinate global units of work, TMF acts as the transaction manager, and an application must use the API provided by TMF to start, commit, and back out global units of work. An application starts a global unit of work by calling BEGINTRANSACTION, and then updates IBM WebSphere MQ resources within the global unit of work by issuing MQPUT, MQPUT1, and MQGET calls within syncpoint control. The application can then commit the global unit of work by calling ENDTRANSACTION, or back it out by calling ABORTTRANSACTION.

An application that is using TMF transactions can only actively work on one transaction at any one time, however using RESUMETRANSACTION allows an application to switch from one active transaction to another, or to being associated with no TMF transaction, without completing or aborting the previously active transaction. Any calls to MQPUT, MQPUT1, or MQGET are made under the currently active TMF transaction, if present, or a local unit of work, if not present. Therefore, care must be taken within the application to ensure that these calls are being made within the correct unit of work.

Within a global unit of work, as well as updating IBM WebSphere MQ resources, an application can update Enscribe files, SQL/MP databases, or SQL/MX databases.

Using global units of work

A global unit of work is implemented as a TMF transaction. An application starts a global unit of work by calling BEGINTRANSACTION, and either commits the unit of work by calling ENDTRANSACTION or backs out the unit of work by calling ABORTTRANSACTION. An application can use other TMF API calls as well.

An application can inherit a TMF transaction from another application. For example, an application (the first application) can perform work within the transaction before replying and passing the transaction back to a second application for further processing. Both the first and the second applications can therefore participate in the same global unit of work that involves updates to IBM WebSphere MQ queues and updates to files and databases. The ability to pass a TMF transaction between applications means that several IBM WebSphere MQ applications can perform messaging operations within the same global unit of work.

An application can manage and control multiple active TMF transactions at the same time. The transactions can be started by the application itself, or inherited from other applications, or both. This means that an application can participate in multiple global units of work at the same time.

The maximum number of concurrent active TMF transactions per process is 1000, which is an architectural limit. If an application is managing multiple TMF transactions, only one transaction can be current at any point in time. Alternatively, none of the transactions can be current. The application can use TMF API calls such as RESUMETRANSACTION, ACTIVATERECEIVETRANSID, and TMF_SET_TX_ID to move the state of being current from one transaction to another, or to designate that no transaction is current. The application uses this level of control to determine whether a messaging operation is performed within a local unit of work, a global unit of work, or outside of syncpoint control:

- If an application calls MQPUT, MQPUT1, or MQGET within syncpoint control when no TMF transaction is current, IBM WebSphere MQ processes the call within a local unit of work.
- If an application calls MQPUT, MQPUT1, or MQGET within syncpoint control when the application has a current TMF transaction, IBM WebSphere MQ processes the call within the global unit of work that is implemented by the current TMF transaction.
- If an application calls MQPUT, MQPUT1, or MQGET outside of syncpoint control, IBM WebSphere MQ processes the call outside of syncpoint control, irrespective of whether the application has a current TMF transaction at the time of the call.

IBM WebSphere MQ never changes the state of an application's TMF transaction during an MQI call, except when a software or hardware failure occurs during processing and IBM WebSphere MQ or the operating system determines that the transaction must be backed out to preserve data integrity. Every MQI call restores the transaction state of the application just before returning control to the application.

Avoiding long running transactions

Avoid designing applications in which TMF transactions remain active for more than a few tens of seconds. Long running transactions can cause the circular audit trail of TMF to fill up. Because TMF is a critical system-wide resource, TMF protects itself by backing out application transactions that are active for too long.

Suppose that the processing within an application is driven by getting messages from a queue, and that the application gets a message from the queue and processes the message within a unit of work. Typically, an application calls MQGET with the wait option and within syncpoint control to get a message from the queue.

If the application is using a global unit of work instead, the specified wait interval on the MQGET call must be short to avoid a long running transaction. This means that the application might need to issue the MQGET call more than once before it retrieves a message.

Installing and uninstalling

Before you start installing IBM WebSphere MQ, consider how you want to use it. Use these topics to help you to prepare for installation, install the product, and verify the installation. There is also information to help you to uninstall the product.

UNIX Linux Windows Completing the following topics in sequence will help you to correctly install and uninstall IBM WebSphere MQ and its components on distributed platforms:
1. Planning your installation
2. Checking requirements
3. Preparing your system
4. Installing components
5. Verifying your installation
6. Uninstalling

You can also apply and remove maintenance to IBM WebSphere MQ. See Maintenance tasks in the Migrating and upgrading section.

Planning your installation

Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

Before you start installing, consider how you want to use IBM WebSphere MQ and review the general planning section. See Planning.

There are a number of choices you must make before you start installing:
- Choosing an installation name
- Making multiple installations
- Choosing a primary installation
- Uninstalling, upgrading, and maintaining the primary installation
- Choosing an installation location
- Choosing what to install
- Installing WebSphere MQ Telemetry
- Windows Planning your installation on Windows systems

When you have finished planning your installation, check your system requirements. See Checking requirements.

Choosing an installation name

Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

You can choose the installation name and make it meaningful to you. For example, you might call a test system *testMQ*.

If you do not specify an installation name when the product is installed, a default installation name is automatically assigned. For the first installation, this name is *Installation1*. For the second installation, the name is *Installation2*, and so on. The installation name *Installation0* is reserved for an installation of IBM WebSphere MQ Version 7.0.1. The installation name cannot be changed after the product is installed.

On UNIX and Linux systems, the first IBM WebSphere MQ installation is automatically given an installation name of *Installation1*. For subsequent installations, you can use the **crtmqinst** command to set the installation name before installing the product.

On Windows systems, you can choose the installation name during the installation process.

The installation name can be up to 16 bytes and must be a combination of alphabetic and numeric characters in the ranges a-z, A-Z, and 0-9. You cannot use blank characters. The installation name must be unique, regardless of whether uppercase or lowercase characters are used. For example, the names `INSTALLATIONNAME` and `InstallationName` are not unique.

You can find out what installation name is assigned to an installation in a particular location using the **dspmqinst** command.

Installation descriptions

Each installation can also have an installation description. This description can give more detailed information about an installation in cases where the installation name cannot provide enough information. These descriptions can be up to 64 single-byte characters, or 32 double-byte characters. The default installation description is blank. You can set the installation description using the **setmqinst** command.

Related concepts:

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

"Choosing an installation location" on page 157
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

Related information:

dspmqinst

setmqinst

crtmqinst

Multiple installations

On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

You can choose where each copy of IBM WebSphere MQ is installed, but each copy must be in a separate installation location. A maximum of 128 installations of IBM WebSphere MQ can exist on a single machine at a time. One installation can be an installation of IBM WebSphere MQ Version 7.0.1 at fix pack 6, or later. You now have a choice:

- Keep the simplicity of maintaining and managing a single installation of IBM WebSphere MQ on a machine.
- Take advantage of the flexibility offered by enabling multiple IBM WebSphere MQ installations.

Before you install multiple copies of IBM WebSphere MQ, you must make several decisions:

- Will you have a copy of IBM WebSphere MQ Version 7.0.1 on the system?

  When IBM WebSphere MQ Version 7.0.1 at fix pack 6, or later, is installed on the system, there are a number of restrictions to consider:
  - On UNIX and Linux systems, IBM WebSphere MQ Version 7.0.1 must be installed in the default location.
  - IBM WebSphere MQ Version 7.0.1 must be the first installation on a system. You cannot install IBM WebSphere MQ Version 7.0.1 after installing version 7.1, or later. If you uninstall version 7.0.1, it cannot be reinstalled while a later version of WebSphere MQ is installed.
  - IBM WebSphere MQ Version 7.0.1 is automatically the primary installation. You cannot select another installation as the primary installation while IBM WebSphere MQ Version 7.0.1 is installed.
- Where will you install each copy of IBM WebSphere MQ?

  You can choose the installation location for your installations at version 7.1, or later. For more information, see "Choosing an installation location" on page 157.
- Do you need a primary installation?

  A primary installation is an installation to which system-wide locations refer. For more information, see "Choosing a primary installation" on page 152.
- How will your applications connect?

  You need to consider how your applications locate the appropriate IBM WebSphere MQ libraries. For more information, see Connecting applications in a multiple installation environment, and Connecting .NET applications in a multiple installation environment.
- Do your existing exits need changing?

  If IBM WebSphere MQ is not installed in the default location, your exits need to be updated. For more information, see Writing and compiling exits and installable services .
- Which queue manager will be associated with which installation?

  Each queue manager is associated with a particular installation. The installation that a queue manager is associated with limits that queue manager so that it can be administered only by commands from that installation. For more information, see Associating a queue manager with an installation.
- How will you set up your environment to work with each installation?

  With multiple installations on a system, you need to consider how you will work with particular installations, and how you will issue commands from that installation. You can either specify the full path to the command, or you can use the **setmqenv** or **crtmqenv** command to set environment variables. Setting the environment variables allows you to omit the path to the commands for that installation. For more information, see setmqenv, and crtmqenv.

When you have answered these questions, you can install IBM WebSphere MQ using the steps provided in "Installing IBM WebSphere MQ" on page 209.

If you have existing installations of IBM WebSphere MQ and you want to use the multiple installation capability to migrate from one version of IBM WebSphere MQ to another version, see Multi-installation queue manager coexistence on UNIX, Linux, and Windows .

The IBM message service client for .NET support pack and multiple installations

For multiple version support, the *Java and .NET Messaging and Web Services* feature must be installed with the IBM WebSphere MQ product. This feature contains all the functionality included in the *IBM Message Service Client for .NET* support pack (IA9H). If the support pack is installed on the system, multiple versions are not supported. You must uninstall the support pack before installing IBM WebSphere MQ. For more information about installing the .NET feature, see Installing WebSphere MQ classes for .NET.

Related information:

Configuring multiple installations
When using multiple installations on the same system, you must configure the installations and queue managers.

Finding installations of WebSphere MQ on a system
If you have multiple IBM WebSphere MQ installations on a system, you can check which versions are installed and where they are.

UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1 to version 7.5

UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1 to version 7.5

Choosing a primary installation

On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Before IBM WebSphere MQ Version 7.1, only one instance of the product could be installed at any time. On Windows systems, several global environment variables were set to point to that installation. On UNIX and Linux systems, symbolic links were added to `/usr/lib`, `/usr/bin`, and `/usr/include`, also pointing at that single installation.

From Version 7.1, you can install multiple versions of IBM WebSphere MQ on UNIX, Linux, and Windows. It is possible to have more than one installation of IBM WebSphere MQ on one of these systems at any time and, optionally, to configure one of these installations as the primary installation. Environment variables and symbolic links pointing to a single installation are less meaningful when multiple versions exist. However, some functions require these system-wide locations to work. For example, custom user scripts for administering IBM WebSphere MQ, and third party products. These functions work only on the primary installation.

On UNIX and Linux systems, if you set an installation as the primary installation, symbolic links to the external libraries and control commands of that installation are added into `/usr/lib`, and `/usr/bin`. If you do not have a primary installation, the symbolic links are not created. For a list of the symbolic links that are made to the primary installation, see "External library and control command links to primary installation on UNIX and Linux" on page 154.

On Windows systems, the global environmental variables point to the directories into which the primary installation was installed. These environment variables are used to locate IBM WebSphere MQ libraries, control commands, and header files. Additionally, on Windows systems, some features of the operating system require the central registration of interface libraries that are then loaded into a single process. With multiple versions of IBM WebSphere MQ, there would be conflicting sets of IBM WebSphere MQ libraries. The features would try to load these conflicting sets of libraries into a single process. Therefore, such features can be used only with the primary installation. For details about some of the features that are limited to use with the primary installation, see "Features that can be used only with the primary installation on Windows" on page 156.

If you have an installation of IBM WebSphere MQ Version 7.0.1 on the system, this installation is automatically the primary installation. The primary installation cannot be changed while Version 7.0.1 is installed. If all the installations on the system are at Version 7.1, or later, you can choose whether to have a primary installation. Consider the options in Table 21 on page 153.

Table 21. Primary installation options.

This table shows the valid installation configurations for primary installations. With a single Version 7.1, or later, it can be either primary, or non-primary. With multiple installations, one at Version 7.0.1 and one or more at Version 7.1, or later,Version 7.0.1 must be the primary, and the other installations must be non-primary. With multiple installations at Version 7.1, or later, one installation can be primary, or all installations can be non-primary.

Options	**Valid installation configurations**		**More information**
	Primary	Non-primary	
Single installation of Version 7.1, or later.	Version 7.1, or later.	None	If you want to continue working with a single installation in the same way as previous releases, configure your installation as the primary installation. For information about this option, see Single installation of IBM WebSphere MQ Version 7.1, or later, configured as the primary installation
	None	Version 7.1, or later.	If you want to continue working with a single installation, but do not want symbolic links or global environment variables created for you, configure your installation as non-primary. For information about the implications of this option, see Single installation of IBM WebSphere MQ Version 7.1, or later, configured as non-primary
Multiple installations: Version 7.0.1 and Version 7.1, or later.	Version 7.0.1	Version 7.1, or later.	If you want to have multiple installations of IBM WebSphere MQ, with one at version 7.0.1, the version 7.0.1 installation is automatically the primary installation. While IBM WebSphere MQ version 7.0.1 is installed, you cannot change which installation is the primary installation. For information about this option and its implications, see Multiple installations of IBM WebSphere MQ, one at Version 7.0.1
Multiple installations: Version 7.1, or later.	Version 7.1, or later.	Version 7.1, or later.	If you want to have multiple installations of WebSphere MQ at version 7.1 or greater, you can choose whether to make one of the installations primary. For information about this option, see Multiple installations of IBM WebSphere MQ Version 7.1, or later
	None	Version 7.1, or later.	

Related concepts:

"Choosing an installation location" on page 157
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

"Choosing an installation name" on page 149
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

Related information:

Single installation of WebSphere MQ Version 7.1, or later, configured as the primary installation
Marking a IBM WebSphere MQ installation as primary adds symbolic links, or global environment variables to the system so that the IBM WebSphere MQ commands and libraries used by applications are automatically available with minimum system setup required.

Single installation of WebSphere MQ Version 7.1, or later, configured as non-primary
If you install IBM WebSphere MQ Version 7.1, or later, as non-primary you might have to configure a library path for applications to load IBM WebSphere MQ libraries. On Windows, some product capabilities are available only when IBM WebSphere MQ is configured as primary.

Multiple installations of WebSphere MQ Version 7.1, or later
You can choose to have one of the IBM WebSphere MQ Version 7.1 installations configured as the primary installation. Your choice depends on how applications locate libraries.

Multiple installations of WebSphere MQ, one at version 7.0.1
IBM WebSphere MQ Version 7.1, or later, can coexist with IBM WebSphere MQ Version 7.0.1 with some limitations.

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

External library and control command links to primary installation on UNIX and Linux

On UNIX and Linux platforms the primary installation is the one to which links from the `/usr` file system are made. However, only a subset of those links created with previous releases are now made.

No links are created from `/usr/include` to any installation and only links to external libraries and documented control commands are made from `/usr/lib` (external libraries) and `/usr/bin` (control commands).

In order to run these commands you must complete the following steps:

1. provide a full path to the command in an available IBM WebSphere MQ installation,
2. use the `setmqenv` script to update your shell environment,
3. manually add the bin directory from a IBM WebSphere MQ installation directory to your PATH,
4. run the **setmqinst** command as root to make one of your existing IBM WebSphere MQ installations the primary installation.

External libraries

Links are made to the following external libraries, both 32-bit and 64-bit:

- libmqm
- libmqm_r
- libmqmxa
- libmqmxa_r
- libmqmax
- libmqmax_r
- libmqmcb
- libmqmcb_r
- libmqic
- libmqic_r
- libmqcxa
- libmqcxa_r
- libmqicb
- libmqicb_r
- libimqb23ia
- libimqb23ia_r
- libimqc23ia
- libimqc23ia_r
- libimqs23ia
- libimqs23ia_r
- libmqmzf

- libmqmzf_r

The following 64-bit only libraries are also linked to:
- libmqmxa64
- libmqmxa64_r
- libmqcxa64
- libmqcxa64_r

Control commands

The following control commands are linked to from `/usr/bin`:
- addmqinf
- amqcrs6a
- amqcrsta
- amqmfsck
- crtmqinst
- dltmqinst
- dspmqinst
- setmqinst
- crtmqcvx
- crtmqm
- dltmqm
- dmpmqaut
- dmpmqlog
- dspmq
- dspmqaut
- dspmqcsv
- dspmqfls
- dspmqinf
- dspmqrte
- dspmqtrc
- dspmqtrn
- dspmqver
- endmqcsv
- endmqlsr
- endmqm
- endmqtrc
- migmbbrk
- rcdmqimg
- rcrmqobj
- rmvmqinf
- rsvmqtrn
- runmqchi
- runmqchl
- runmqckm
- runmqdlq

- runmqlsr
- runmqsc
- runmqtmc
- runmqtrm
- setmqaut
- setmqenv
- setmqm
- setmqprd
- strmqbrk
- strmqcsv
- strmqikm
- strmqm
- strmqtrc

Related concepts:

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

"Features that can be used only with the primary installation on Windows"
Some Windows operating-system features can be used only with the primary installation. This restriction is due to the central registration of interface libraries, which might conflict as a result of multiple versions of IBM WebSphere MQ being installed.

Features that can be used only with the primary installation on Windows

Some Windows operating-system features can be used only with the primary installation. This restriction is due to the central registration of interface libraries, which might conflict as a result of multiple versions of IBM WebSphere MQ being installed.

The .NET monitor

The IBM WebSphere MQ .NET monitor can run in two different modes: transactional and non-transactional. The transactional mode uses MSDTC transaction coordination and requires that the .NET monitor is registered with COM+. The .NET monitor from the primary installation is the only .NET monitor that is registered with COM+.

Any attempt to run the .NET monitor in transactional mode with a non-primary installation results in the failure of the .NET monitor to enlist with MSDTC. The .NET monitor receives an `MQRC_INSTALLATION_MISMATCH` error, which in turn results in an AMQ8377 error message on the console.

COM/ActiveX interface classes

The COM/ActiveX interface classes are registered only for the primary installation. If there is an installation of IBM WebSphere MQ Version 7.0.1 on the system, the COM/ActiveX interface classes registered are not capable of connecting to queue managers running under other installations. If the primary installation is an installation of IBM WebSphere MQ Version 7.1 or later, the interface classes can connect to queue managers associated with any installation. Server COM/ActiveX applications are limited by this restriction, but client applications can connect to any queue manager.

Any attempt to start a COM/ActiveX application that uses libraries from installations other than the primary installation results in failure with an `MQRC_Q_MGR_NOT_AVAILABLE` error.

Related concepts:

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

"External library and control command links to primary installation on UNIX and Linux" on page 154
On UNIX and Linux platforms the primary installation is the one to which links from the /usr file system are made. However, only a subset of those links created with previous releases are now made.

Uninstalling, upgrading, and maintaining the primary installation

On all platforms, if you uninstall the primary installation, it ceases to be the primary installation. You must run the **setmqinst** command to select a new primary installation. On Windows, if you update the primary installation, it continues to be the primary installation. If you apply a fix pack to the primary installation, it continues to be the primary installation.

Be cautious about the effect uninstalling or upgrading the primary installation has on applications. Applications might be using the linkage library of the primary installation to switch to the linkage library of another installation. If such an application is running, you might not be able to uninstall the primary installation. The operating system might have locked the link library of the primary installation on behalf of the application. If the primary installation has been uninstalled, an application that loads the IBM WebSphere MQ libraries it requires by linking to the primary installation is not able to start.

The solution is to switch the primary installation to another installation before uninstalling. Stop, and restart applications that are linked through the previous primary installation before uninstalling it.

Windows

If you update the primary installation, it stops being the primary installation at the beginning of the update procedure. If, by the end of the update procedure, you have not made another installation primary, the upgraded installation is made primary again.

Maintenance

If you apply a fix pack to the primary installation, it stops being the primary installation at the beginning of the maintenance procedure. If, by the end of the maintenance procedure, you have not made another installation primary, the upgraded installation is made primary again.

Related concepts:

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

"Uninstalling" on page 304
The topics in this section provide instructions on how to uninstall components.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

WebSphere MQ maintenance tasks
This section describes how to apply and remove maintenance to IBM WebSphere MQ.

Choosing an installation location

You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

The default location for the IBM WebSphere MQ product code is shown in the following table:

Table 22. Installation location of IBM WebSphere MQ

Platform	**Installation Location**
Linux, HP-UX, and Solaris	`/opt/mqm`
AIX	`/usr/mqm`
Windows 32-bit	`C:\Program Files\IBM\WebSphere MQ`
Windows 64-bit	`C:\Program Files (x86)\IBM\WebSphere MQ`

On UNIX and Linux systems, working data is stored in `/var/mqm`, but you cannot change this location; for more information about the directory structure of UNIX and Linux systems, see Directory structure on UNIX and Linux systems .

Custom location installation

For an installation into a custom location, the path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes. On UNIX and Linux systems, the path must not contain spaces.

- On AIX, the product is installed into a User Specified Install Location (USIL), which can be either an existing USIL or a new USIL that is automatically created by the installation process. If a custom location is specified, the product location is the path specified during installation, plus `/usr/mqm`.

  For example, the path specified is `/usr/custom_location`. The *`MQ_INSTALLATION_PATH`* is `/usr/custom_location/usr/mqm`.

  Access permissions for the USIL directory should be set to rwx for user and r-x for group and others (755).
- On Windows, Linux, HP-UX, and Solaris, the product location is the same path as specified during installation.

  For example, on Linux, the path specified is `/opt/custom_location`. The *`MQ_INSTALLATION_PATH`* is `/opt/custom_location`.

For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location.

For more information about how to install to a custom location, see the installation topics for the appropriate platform.

Additional location restrictions

New WebSphere MQ installations should not be located in the following paths:

- In a path that is a subdirectory of another existing installation.
- In a path that is part of the direct path to an existing installation.
- In a path that is a subdirectory of the default location, for example:
  - /usr/mqm on AIX
  - /opt/mqm on Linux, Solaris and HP-UX platforms
- In a directory or subdirectory that is, or might later be used by another product, for example, an IBM DB2® installation, or operating system component.

An installation should not be located in /opt/mqm/v80, /opt/mqm/v75, /opt/mqm/inst2/mq71, or other directory located under /opt/mqm on Linux, Solaris and HP-UX platforms.

If WebSphere MQ is installed in /opt/IBM/MQ/installations/1, you can not install in /opt/IBM/MQ/installations/1/a. Additionally, you should not install a new installation to /opt/IBM/MQ. However, you can install a new installation in /opt/IBM/MQ/installations/2 or /opt/IBM/MQnew because neither of these is a part of the direct path /opt/IBM/MQ/installations/1.

You must not install to any directory located under /opt/IBM/db2.

The reason an installation should not be located in a path that is a subdirectory of the default location is to avoid the risk if you later decide to install WebSphere MQ into the default location, and cannot then do so. If you do subsequently install into the default location, because WebSphere MQ has full access rights over the installation directory, existing files might be replaced or deleted. Scripts that you might subsequently run to uninstall WebSphere MQ might remove the installation directory at the end of the script.

Related concepts:

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

"Choosing an installation name" on page 149
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

"Choosing what to install"
You can select the components or features that you require when you install IBM WebSphere MQ.

Related tasks:

"Installing a WebSphere MQ server" on page 209
After preparing your system for installation you may install IBM WebSphere MQ by following the appropriate instructions for your platform. After installation, you might want to verify your installation to check that installation has been successful.

Choosing what to install

You can select the components or features that you require when you install IBM WebSphere MQ.

IBM WebSphere MQ can be installed as a server or a client. The installation images can be downloaded, or IBM WebSphere MQ can be installed from a DVD.

A IBM WebSphere MQ server is an installation of one or more queue managers that provide queueing services to one or more clients. All the IBM WebSphere MQ objects, for example queues, exist only on the queue manager machine (the IBM WebSphere MQ server machine), and not the client. A IBM WebSphere MQ server can also support local IBM WebSphere MQ applications. To install a IBM WebSphere MQ server see, "Installing a WebSphere MQ server" on page 209.

A IBM WebSphere MQ MQI client is a component that allows an application running on one system to communicate with a queue manager running on another system. The output from the call is sent back to the client, which passes it back to the application. To install a IBM WebSphere MQ MQI client see, Installing a IBM WebSphere MQ client.

It is possible to have both a server and a client installation on the same system. See, "Installing a IBM WebSphere MQ client" on page 256.

IBM WebSphere MQ Advanced Message Security is a separately installed and licensed component of WebSphere MQ and is another option on the IBM WebSphere MQ installer. To install IBM WebSphere MQ Advanced Message Security, see "Installing WebSphere MQ Advanced Message Security" on page 277.

For detailed explanations of all the components that you can install, see the following platform-specific topics:

- "IBM WebSphere MQ components for AIX systems"
- "IBM WebSphere MQ components for HP-UX systems" on page 163
- "IBM WebSphere MQ components for Linux systems" on page 166
- "Installing IBM WebSphere MQ server on Linux Ubuntu" on page 220
- "IBM WebSphere MQ components for Solaris systems" on page 169
- "IBM WebSphere MQ features for Windows systems" on page 173

Related concepts:

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

"Choosing an installation location" on page 157
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

"Choosing an installation name" on page 149
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

IBM WebSphere MQ components for AIX systems

You can select the components that you require when you install IBM WebSphere MQ.

Table 23 shows the components that are available when installing a IBM WebSphere MQ server or client on an AIX system:

Table 23. IBM WebSphere MQ components for AIX systems.

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Runtime	Contains files that are common to both server and client installations. **Note:** This component must be installed.	✓	✓	✓	mqm.base.runtime
Server	You can use the server to run queue managers on your system and connect to other systems over a network. Provides messaging and queuing services to applications, and support for IBM WebSphere MQ client connections.	✓		✓	mqm.server

Table 23. IBM WebSphere MQ components for AIX systems (continued).

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Standard Client	The IBM WebSphere MQ MQI client is a small subset of IBM WebSphere MQ, without a queue manager, that uses the queue manager and queues on other (server) systems. It can be used only when the system it is on is connected to another system that is running a full server version of IBM WebSphere MQ. The client and the server can be on the same system if required.	✓	✓	✓	mqm.client.rte
SDK	The SDK is required for compiling applications. It includes sample source files, and the bindings (files .H, .LIB, .DLL, and others), that you need to develop applications to run on IBM WebSphere MQ.	✓	✓	✓	mqm.base.sdk
Sample programs	The sample application programs are needed if you want to check your IBM WebSphere MQ installation using the verification procedures.	✓	✓	✓	mqm.base.samples
Java messaging	The files needed for messaging using Java (includes Java Messaging Service).	✓	✓	✓	mqm.java.rte
Man pages	UNIX man pages, in U.S. English, for: control commands MQI commands MQSC commands	✓	✓	✓	mqm.man.en_US.data
Java JRE	A Java Runtime Environment, Version 6.0, that is used by those parts of IBM WebSphere MQ that are written in Java.	✓	✓	✓	mqm.jre.rte
Message Catalogs	For available languages, see the table of message catalogs that follows.	✓	✓	✓	
IBM Global Security Kit	IBM Global Security Kit V8 Certificate and SSL Base Runtime.	✓	✓	✓	mqm.gskit.rte

Table 23. IBM WebSphere MQ components for AIX systems (continued).

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
MQ Telemetry	MQ Telemetry supports the connection of Internet Of Things (IOT) devices (that is, remote sensors, actuators and telemetry devices) that use the MQ Telemetry Transport (MQTT) protocol. The MQ Telemetry component comprises: • The telemetry service. This is also know as the MQ Extended Reach (MQXR) service. It enables a queue manager to act as an MQTT server, and communicate with MQTT client apps. • A set of MQTT client libraries. These libraries help you write the MQTT client apps that IOT devices use to communicate with MQTT servers. The most up-to-date version of the MQTT client libraries is available in the free download Mobile Messaging and M2M Client Pack. See also "Installing IBM WebSphere MQ Telemetry" on page 179.	✓	✓	✓	mqm.xr.service mqm.xr.clients
Managed File Transfer	MQ Managed File Transfer transfers files between systems in a managed and auditable way, regardless of file size or the operating systems used. For information about the function of each component, see WebSphere MQ Managed File Transfer product options.	✓		✓	mqm.ft.agent mqm.ft.base mqm.ft.logger mqm.ft.service mqm.ft.tools
Advanced Message Security	Provides a high level of protection for sensitive data flowing through the IBM WebSphere MQ network, while not impacting the end applications. You must install this component on all IBM WebSphere MQ installations that host queues you want to protect. You must install the IBM Global Security Kit component on any IBM WebSphere MQ installation that is used by a program that puts or gets messages to or from a protected queue, unless you are using only Java client connections.	✓		✓	mqm.ams.rte

Table 24. IBM WebSphere MQ message catalogs for AIX systems.

A two-column table listing the available message catalogs.

Message catalog language	Component name
Brazilian Portuguese	mqm.msg.pt_BR
Czech	mqm.msg.cs_CZ
French	mqm.msg.fr_FR
German	mqm.msg.de_DE
Hungarian	mqm.msg.hu_HU
Italian	mqm.msg.it_IT
Japanese	mqm.msg.ja_JP, mqm.msg.Ja_JP
Korean	mqm.msg.ko_KR
Polish	mqm.msg.pl_PL
Russian	mqm.msg.ru_RU
Spanish	mqm.msg.es_ES
Simplified Chinese	mqm.msg.zh_CN, mqm.msg.Zh.CN
Traditional Chinese	mqm.msg.zh_TW, mqm.msg.Zh_TW
U.S. English	mqm.msg.en_US

Related concepts:

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

IBM WebSphere MQ client components for HP Integrity NonStop Server

There are no optional components within the client installer when you install the IBM WebSphere MQ client for HP Integrity NonStop Server.

An installation of the IBM WebSphere MQ client for HP Integrity NonStop Server contains product binary files, command utilities, and samples.

Related concepts:

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

IBM WebSphere MQ components for HP-UX systems

You can select the components that you require when you install IBM WebSphere MQ.

Table 25 on page 164 shows the components that are available when installing a IBM WebSphere MQ server or client on an HP-UX system:

Table 25. IBM WebSphere MQ components for HP-UX systems.

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Runtime	Contains files that are common to both server and client installations. **Note:** This component must be installed.	✓	✓		MQSERIES.MQM-RUNTIME
Server	You can use the server to run queue managers on your system and connect to other systems over a network. Provides messaging and queuing services to applications, and support for IBM WebSphere MQ client connections.	✓			MQSERIES.MQM-SERVER
Standard Client	The IBM WebSphere MQ MQI client is a small subset of IBM WebSphere MQ, without a queue manager, that uses the queue manager and queues on other (server) systems. It can be used only when the system it is on is connected to another system that is running a full server version of IBM WebSphere MQ. The client and the server can be on the same system if required.	✓	✓		MQSERIES.MQM-CL-HPUX
SDK	The SDK is required for compiling applications. It includes sample source files, and the bindings (files .H, .LIB, .DLL, and others), that you need to develop applications to run on IBM WebSphere MQ.	✓	✓		MQSERIES.MQM-BASE
Sample programs	The sample application programs are needed if you want to check your IBM WebSphere MQ installation using the verification procedures.	✓	✓		MQSERIES.MQM-SAMPLES
Java messaging	The files needed for messaging using Java (includes Java Messaging Service).	✓	✓		MQSERIES.MQM-JAVA
Man pages	UNIX man pages, in U.S. English, for: control commands MQI commands MQSC commands	✓	✓		MQSERIES.MQM-MAN
Java JRE	A Java Runtime Environment, Version 6.0, that is used by those parts of IBM WebSphere MQ that are written in Java.	✓	✓		MQSERIES.MQM-JAVAJRE
Message Catalogs	For available languages, see the table of message catalogs that follows.	✓	✓		

Table 25. IBM WebSphere MQ components for HP-UX systems (continued).

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	HypervisComponent name	
IBM Global Security Kit	IBM Global Security Kit V8 Certificate and SSL Base Runtime.	✓	✓		MQSERIES.MQM-GSKIT
Managed File Transfer	MQ Managed File Transfer transfers files between systems in a managed and auditable way, regardless of file size or the operating systems used. For information about the function of each component, see WebSphere MQ Managed File Transfer product options.	✓			MQSERIES.MQM-FTAGENT MQSERIES.MQM-FTBASE MQSERIES.MQM-FTLOGGER MQSERIES.MQM-FTSERVICE MQSERIES.MQM-FTTOOLS
Advanced Message Security	Provides a high level of protection for sensitive data flowing through the IBM WebSphere MQ network, while not impacting the end applications. You must install this component on all IBM WebSphere MQ installations that host queues you want to protect. You must install the IBM Global Security Kit component on any IBM WebSphere MQ installation that is used by a program that puts or gets messages to or from a protected queue, unless you are using only Java client connections.	✓			MQSERIES.MQM-AMS

Table 26. IBM WebSphere MQ message catalogs for HP-UX systems.

A two-column table listing the available message catalogs.

Message catalog language	Component name
Brazilian Portuguese	MQSERIES.MQM-MC-PORT
Czech	MQSERIES.MQM-MC-CZECH
French	MQSERIES.MQM-MC-FRENCH
German	MQSERIES.MQM-MC-GERMAN
Hungarian	MQSERIES.MQM-MC-HUNGARIAN
Italian	MQSERIES.MQM-MC-ITALIAN
Japanese	MQSERIES.MQM-MC-JAPAN
Korean	MQSERIES.MQM-MC-KOREAN
Polish	MQSERIES.MQM-MC-POLISH
Russian	MQSERIES.MQM-MC-RUSSIAN
Spanish	MQSERIES.MQM-MC-SPANISH
Simplified Chinese	MQSERIES.MQM-MC-CHINES
Traditional Chinese	MQSERIES.MQM-MC-CHINET

Table 26. IBM WebSphere MQ message catalogs for HP-UX systems (continued).

A two-column table listing the available message catalogs.

Message catalog language	Component name
U.S. English	not applicable

Related concepts:

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

IBM WebSphere MQ components for Linux systems

You can select the components that you require when you install IBM WebSphere MQ.

Table 27 shows the components that are available when installing a IBM WebSphere MQ server or client on a Linux system:

Table 27. IBM WebSphere MQ components for Linux systems.

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Runtime	Contains files that are common to both server and client installations. **Note:** This component must be installed.	✓	✓	✓	MQSeriesRuntime
Server	You can use the server to run queue managers on your system and connect to other systems over a network. Provides messaging and queuing services to applications, and support for IBM WebSphere MQ client connections.	✓		✓	MQSeriesServer
Standard Client	The IBM WebSphere MQ MQI client is a small subset of IBM WebSphere MQ, without a queue manager, that uses the queue manager and queues on other (server) systems. It can be used only when the system it is on is connected to another system that is running a full server version of IBM WebSphere MQ. The client and the server can be on the same system if required.	✓	✓	✓	MQSeriesClient
SDK	The SDK is required for compiling applications. It includes sample source files, and the bindings (files .H, .LIB, .DLL, and others), that you need to develop applications to run on IBM WebSphere MQ.	✓	✓	✓	MQSeriesSDK

Table 27. IBM WebSphere MQ components for Linux systems (continued).

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Sample programs	The sample application programs are needed if you want to check your IBM WebSphere MQ installation using the verification procedures.	✓	✓	✓	MQSeriesSamples
Java messaging	The files needed for messaging using Java (includes Java Messaging Service).	✓	✓	✓	MQSeriesJava
Man pages	UNIX man pages, in U.S. English, for: control commands MQI commands MQSC commands	✓	✓	✓	MQSeriesMan
Java JRE	A Java Runtime Environment, Version 6.0, that is used by those parts of IBM WebSphere MQ that are written in Java.	✓	✓	✓	MQSeriesJRE
Message Catalogs	For available languages, see the table of message catalogs that follows.	✓	✓	✓	
IBM Global Security Kit	IBM Global Security Kit V8 Certificate and SSL Base Runtime.	✓	✓	✓	MQSeriesGSKit
MQ Telemetry	MQ Telemetry supports the connection of Internet Of Things (IOT) devices (that is, remote sensors, actuators and telemetry devices) that use the MQ Telemetry Transport (MQTT) protocol. The MQ Telemetry component comprises: • The telemetry service. This is also known as the MQ Extended Reach (MQXR) service. It enables a queue manager to act as an MQTT server, and communicate with MQTT client apps. • A set of MQTT client libraries. These libraries help you write the MQTT client apps that IOT devices use to communicate with MQTT servers. MQ Telemetry is only available on Linux for System x (64 bit) and Linux for System z. The most up-to-date version of the MQTT client libraries is available in the free download Mobile Messaging and M2M Client Pack. See also "Installing IBM WebSphere MQ Telemetry" on page 179.	✓	✓	✓	MQSeriesXRService MQSeriesXRClients

Table 27. IBM WebSphere MQ components for Linux systems (continued).

A six-column table listing the available product components, what source they are available from, and the name of the component.

Component	Description	Server DVD	Client DVD	Hyperviso	Component name
MQ Explorer	Use IBM WebSphere MQ MQ Explorer to administer and monitor resources on Linux x86 and x86-64 systems.	✓		✓	MQSeriesExplorer
Managed File Transfer	MQ Managed File Transfer transfers files between systems in a managed and auditable way, regardless of file size or the operating systems used. For information about the function of each component, see WebSphere MQ Managed File Transfer product options.	✓		✓	MQSeriesFTAgent MQSeriesFTBase MQSeriesFTLogger MQSeriesFTService MQSeriesFTTools
Advanced Message Security	Provides a high level of protection for sensitive data flowing through the IBM WebSphere MQ network, while not impacting the end applications. You must install this component on all IBM WebSphere MQ installations that host queues you want to protect. You must install the IBM Global Security Kit component on any IBM WebSphere MQ installation that is used by a program that puts or gets messages to or from a protected queue, unless you are using only Java client connections.	✓		✓	MQSeriesAMS

Table 28. IBM WebSphere MQ message catalogs for Linux systems.

A two-column table listing the available message catalogs.

Message catalog language	Component name
Brazilian Portuguese	MQSeriesMsg_pt
Czech	MQSeriesMsg_cs
French	MQSeriesMsg_fr
German	MQSeriesMsg_de
Hungarian	MQSeriesMsg_hu
Italian	MQSeriesMsg_it
Japanese	MQSeriesMsg_ja
Korean	MQSeriesMsg_ko
Polish	MQSeriesMsg_pl
Russian	MQSeriesMsg_ru
Spanish	MQSeriesMsg_es
Simplified Chinese	MQSeriesMsg_Zh_CN
Traditional Chinese	MQSeriesMsg_Zh_TW

Table 28. IBM WebSphere MQ message catalogs for Linux systems (continued).

A two-column table listing the available message catalogs.

Message catalog language	Component name
U.S. English	not applicable

Related concepts:

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

IBM WebSphere MQ components for Solaris systems

You can select the components that you require when you install IBM WebSphere MQ.

Table 29 shows the components that are available when installing a IBM WebSphere MQ server or client on a Solaris system.

Note: When you install interactively on Solaris systems, the options that are available install various combinations of the components listed in this table. Details are given in the "Interactive installation" on page 171 section.

Table 29. IBM WebSphere MQ components for Solaris systems.

A six-column table listing the available product components, what source they are available from, and the names of the components.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Runtime	Contains files that are common to both server and client installations. **Note:** This component must be installed.	✓	✓		runtime
Server	You can use the server to run queue managers on your system and connect to other systems over a network. Provides messaging and queuing services to applications, and support for IBM WebSphere MQ client connections.	✓			server
Standard Client	The IBM WebSphere MQ MQI client is a small subset of IBM WebSphere MQ, without a queue manager, that uses the queue manager and queues on other (server) systems. It can be used only when the system it is on is connected to another system that is running a full server version of IBM WebSphere MQ. The client and the server can be on the same system if required.	✓	✓		sol_client
SDK	The SDK is required for compiling applications. It includes sample source files, and the bindings (files .H, .LIB, .DLL, and others), that you need to develop applications to run on IBM WebSphere MQ.	✓	✓		base

Table 29. IBM WebSphere MQ components for Solaris systems (continued).

A six-column table listing the available product components, what source they are available from, and the names of the components.

Component	Description	Server DVD	Client DVD	Hypervisor	Component name
Sample programs	The sample application programs are needed if you want to check your IBM WebSphere MQ installation using the verification procedures.	✓	✓		samples
Java messaging	The files needed for messaging using Java (includes Java Messaging Service).	✓	✓		java
Man pages	UNIX man pages, in U.S. English, for: control commands MQI commands MQSC commands	✓	✓		man
Java JRE	A Java Runtime Environment, Version 6.0, that is used by those parts of IBM WebSphere MQ that are written in Java.	✓	✓		jre
Message Catalogs	For available languages, see the table of message catalogs that follows.	✓	✓		
IBM Global Security Kit	IBM Global Security Kit V8 Certificate and SSL Base Runtime.	✓	✓		gskit
Managed File Transfer	MQ Managed File Transfer transfers files between systems in a managed and auditable way, regardless of file size or the operating systems used. For information about the function of each component, see WebSphere MQ Managed File Transfer product options.	✓			ftagent ftbase ftlogger ftservice fttools
Advanced Message Security	Provides a high level of protection for sensitive data flowing through the IBM WebSphere MQ network, while not impacting the end applications. You must install this component on all IBM WebSphere MQ installations that host queues you want to protect. You must install the IBM Global Security Kit component on any IBM WebSphere MQ installation that is used by a program that puts or gets messages to or from a protected queue, unless you are using only Java client connections.	✓			mqams

Table 30. IBM WebSphere MQ message catalogs for Solaris systems.

A two-column table listing the available message catalogs.

Message catalog language	Component name
Brazilian Portuguese	Pt_BR
Czech	Cs_CZ
French	Fr_FR
German	De_DE
Hungarian	Hu_HU
Italian	It_IT
Japanese	Ja_JP
Korean	Ko_KR
Polish	Pl_PL
Russian	Ru_RU
Spanish	Es_ES
Simplified Chinese	Zh_CN
Traditional Chinese	Zh_TW
U.S. English	not applicable

Interactive installation

The options available with interactive installation install various combinations of the product components described in the previous tables. The following table shows you what will be installed for each option, together with the option number on the server and client DVDs:

Table 31. IBM WebSphere MQ interactive installation options for Solaris systems.

A four-column table listing interactive installation options and the components installed with each one. Server and client option numbers are also listed.

Interactive installation option	Components installed	Server DVD option number	Client DVD option number
IBM WebSphere MQ Server	base runtime server java gskit	1	
Man pages	runtime man	2	1
Sample programs	base runtime samples	3	2
IBM WebSphere MQ MQI client libraries (including Java, JMS, and Web Services support)	base runtime sol_client java gskit	4	3
IBM Java runtime for Solaris, Java 2 Technology Edition, Version 6	jre runtime	5	

Table 31. IBM WebSphere MQ interactive installation options for Solaris systems (continued).

A four-column table listing interactive installation options and the components installed with each one. Server and client option numbers are also listed.

Interactive installation option	Components installed	Server DVD option number	Client DVD option number
IBM Global Security Kit for IBM WebSphere MQ	gskit jre runtime	6	
IBM WebSphere MQ Managed File Transfer Service	ftservice ftbase jre java runtime ftagent	7	
IBM WebSphere MQ Managed File Transfer Tools	fttools ftbase jre java runtime	8	
IBM WebSphere MQ Managed File Transfer Agent	ftagent ftbase jre java runtime	9	
IBM WebSphere MQ Managed File Transfer Logger	ftlogger ftbase jre java runtime server	10	
WebSphere MQ Advanced Message Security	runtime mqams	11	
Spanish message catalog	runtime Es_ES	12	4
French message catalog	runtime Fr_FR	13	5
German message catalog	runtime De_DE	14	6
Japanese message catalog	runtime Ja_JP	15	7
Italian message catalog	runtime It_IT	16	8
Brazilian Portuguese message catalog	runtime Pt_BR	17	9
Traditional Chinese message catalog	runtime Zh_TW	18	10
Simplified Chinese message catalog	runtime Zh_CN	19	11
Korean message catalog	runtime Ko_KR	20	12

Table 31. IBM WebSphere MQ interactive installation options for Solaris systems (continued).

A four-column table listing interactive installation options and the components installed with each one. Server and client option numbers are also listed.

Interactive installation option	Components installed	Server DVD option number	Client DVD option number
Russian message catalog	runtime Ru_RU	21	13
Hungarian message catalog	runtime Hu_HU	22	14
Polish message catalog	runtime Pl_PL	23	15
Czech message catalog	runtime Cs_CZ	24	16

Related concepts:

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

IBM WebSphere MQ features for Windows systems

You can select the features that you require when you install IBM WebSphere MQ.

The following table shows the features that are available when installing a IBM WebSphere MQ server or client on a Windows system.

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
Server	Server	You can use the server to run queue managers on your system and connect to other systems over a network. Provides messaging and queuing services to applications, and support for IBM WebSphere MQ client connections.	✓	
MQ Explorer	Explorer	IBM WebSphere MQ Explorer allows you to administer and monitor resources in IBM WebSphere MQ.	✓	

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
Managed File Transfer Service	MFT_Service	The IBM WebSphere MQ Managed File Transfer Service install option installs a file transfer agent that has additional capabilities beyond those provided by the file transfer agent installed via the IBM WebSphere MQ Managed File Transfer Agent install option. These additional capabilities are:- • Create protocol bridge agents which are used to send and receive files with legacy FTP, FTPS or SFTP servers • Deploy the Web Gateway feature which provides RESTful interfaces for building web applications that transfer files The IBM WebSphere MQ Managed File Transfer Service install option must be installed on systems where the IBM WebSphere MQ Server install option is already installed.	✓	

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
Managed File Transfer Logger	MFT_Logger	The IBM WebSphere MQ Managed File Transfer Logger install option installs a file transfer logger which connects to a IBM WebSphere MQ queue manager, often the queue manager designated as the coordination queue manager. It logs file transfer audit related data to either a database or a file. It must be installed on systems where the IBM WebSphere MQ Server install option is already installed.	✓	
Managed File Transfer Agent	MFT_Agent	The IBM WebSphere MQ Managed File Transfer Agent install option installs a file transfer agent which connects to a IBM WebSphere MQ queue manager and transfers file data, as messages, to other file transfer agents. These must be installed either as part of the IBM WebSphere MQ Managed File Transfer Agent or IBM WebSphere MQ Managed File Transfer Service install options.	✓	

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
Managed File Transfer Tools	MFT_Tools	The IBM WebSphere MQ Managed File Transfer Tools install option installs command line tools that are used to interact with file transfer agents. You can use these tools to start file transfers, schedule file transfers and create resource monitors from the command line. The IBM WebSphere MQ Managed File Transfer Tools can be installed and used on either a system where file transfer agents are installed, or on a system where no file transfer agents are installed.	✓	
Windows Client	Client	The IBM WebSphere MQ client is a small subset of IBM WebSphere MQ, without a queue manager, that uses the queue manager and queues on other (server) systems. It can be used only when the system it is on is connected to another system that is running a full server version of IBM WebSphere MQ. The client and server can be on the same system if required.	✓	✓
Java and .NET Messaging and Web Services	JavaMsg	The files needed for messaging using Java (includes Java Message Service support) and IBM WebSphere MQ Web Services.	✓	✓

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
Development Toolkit	Toolkit	This feature includes sample source files, and the bindings (files .H, .LIB, .DLL, and others), that you need to develop applications to run on IBM WebSphere MQ . Bindings and samples are provided for the following languages: C, C++, Visual Basic, ActiveX, Cobol, and .NET (including C#). Java and Java Message Service support is included and samples are provided for MTS (COM+), and MQSC.	✓	✓

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
MQ Telemetry	XR_Service XR_Clients	MQ Telemetry supports the connection of Internet Of Things (IOT) devices (that is, remote sensors, actuators and telemetry devices) that use the MQ Telemetry Transport (MQTT) protocol. The MQ Telemetry component comprises: • The telemetry service. This is also know as the MQ Extended Reach (MQXR) service. It enables a queue manager to act as an MQTT server, and communicate with MQTT client apps. • A set of MQTT client libraries. These libraries help you write the MQTT client apps that IOT devices use to communicate with MQTT servers. The most up-to-date version of the MQTT client libraries is available in the free download Mobile Messaging and M2M Client Pack. See also "Installing IBM WebSphere MQ Telemetry" on page 179.	✓	✓

A five-column table listing the available product components on Windows systems and what source they are available from.

Interactive displayed name	Non-interactive displayed name	Description	Server DVD	Client DVD
Advanced Message Security	AMS	Provides a high level of protection for sensitive data flowing through the IBM WebSphere MQ network, while not impacting the end applications. You must install this component on all IBM WebSphere MQ installations that host queues you want to protect. You must install the IBM Global Security Kit component on any IBM WebSphere MQ installation that is used by a program that puts or gets messages to or from a protected queue, unless you are using only Java client connections.	✓	

Related concepts:

"Choosing what to install" on page 159
You can select the components or features that you require when you install IBM WebSphere MQ.

"Planning your installation" on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

Installing IBM WebSphere MQ Telemetry

From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

If IBM WebSphere MQ Version 7.0.1 is installed with the IBM WebSphere MQ Telemetry plug-in, you must migrate the installation to IBM WebSphere MQ Version 7.5. See Migrating IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5 for information and instructions about this process.

To run WebSphere MQ Extended Reach (MQXR) service, you need the following software components:

- A Java Runtime Environment (JRE), for each system on which you run applications.
- WebSphere MQ Java

IBM WebSphere MQ Telemetry clients

The IBM WebSphere MQ Telemetry installer creates a directory called `mqxr` in which all IBM

WebSphere MQ Telemetry components are installed. Two IBM WebSphere MQ Telemetry clients, called basic and advanced, are automatically installed in the `mqxr/SDK` sub-folder. For Windows, the clients by default are installed in `Program Files\IBM\WebSphere MQ\mqxr\SDK`. For Linux and AIX, the clients by default are installed in `/opt/mqm/mqxr/SDK`.

IBM WebSphere MQ Telemetry runtime

IBM WebSphere MQ Telemetry runtime is an extension to theIBM WebSphere MQ queue manager. The IBM WebSphere MQ Telemetry feature supports the connection of telemetry devices from the edge of a network to IBM WebSphere MQ. This connection is made possible by the IBM WebSphere MQ Telemetry Transport (MQTT) protocol.

IBM WebSphere MQ Telemetry runtime can be configured using the IBM WebSphere MQ Explorer. For a queue manager to accept connections from a telemetry device, one or more telemetry channels are needed. To enable MQTT there is a define sample configuration wizard that can be run from IBM WebSphere MQ Explorer. The wizard runs through a series of steps including defining and starting the telemetry (MQXR) service, setting up the default transmission queue, and configuring a telemetry channel. For more information about using the define sample configuration wizard, and any implications, see "Verifying the installation of IBM WebSphere MQ Telemetry by using IBM WebSphere MQ Explorer" on page 300.

Support for IBM WebSphere MQ Explorer

You can use IBM WebSphere MQ Explorer to configure and manage the IBM WebSphere MQ Telemetry runtime component. The extension provides the following capabilities:

- Telemetry node and content panel - providing welcome information, define sample configuration wizard, run MQTT client utility, Help on IBM WebSphere MQ Telemetry, and status information about the IBM WebSphere MQ Telemetry Service.
- Define sample configuration wizard - quickly configures a queue manager to support MQTT.
- New Telemetry Channel wizard - gathers information required to create a telemetry channel object.
- Telemetry Channels node and content panel - displays telemetry channels in the IBM WebSphere MQ Explorer Content view.
- Telemetry Channel Status node and content panel - displays telemetry channel status in the IBM WebSphere MQ Explorer Content view.
- MQTT Client Utility - provides a simple GUI for publishing and subscribing to topics.
- Help on IBM WebSphere MQ Telemetry.

You can administer IBM WebSphere MQ Telemetry Version 7.0.1 only from the Version 7.0.1 IBM WebSphere MQ Explorer. If you connect the Version 7.5 explorer remotely to a Version 7.0.1 queue manager, no telemetry resources are displayed. You cannot connect a Version 7.5 IBM WebSphere MQ Explorer locally to a Version 7.0.1 queue manager on the same server.

IBM WebSphere MQ Telemetry client libraries and SDK

You can develop your applications using the IBM WebSphere MQ Telemetry client libraries and the development SDK. The client libraries and the development SDK can be imported into a development environment (for example WebSphere Eclipse Platform). After relevant applications are developed, these applications, and client libraries can then be deployed together to the appropriate system. The SDK includes the following features:

- Client documentation.
- MQTT client libraries
  - Java MQTT libraries
  - Native C MQTT libraries
- Advanced function components
  - MQ Telemetry Daemon for Devices

Two copies of the `com.ibm.micro.client.mqttv3.jar` JAR file are installed. One copy has a version number as part of the file name. For example: `com.ibm.micro.client.mqttv3_3.0.2.0-20100723.jar`. Use the versioned copy in OSGi applications. The content of the JAR files is the same.

The IBM WebSphere MQ Telemetry Server and Client components can be installed together, or you can install each component on a separate system. You can install the IBM WebSphere MQ Telemetry runtime component on one system and configure and manage it using the IBM WebSphere MQ Explorer installed on another system. However, the components can be installed only on systems with the appropriate prerequisites. If you have the prerequisites for the IBM WebSphere MQ Telemetry runtime on your system, then both components are installed. There is an option to select not to install the IBM WebSphere MQ Telemetry client libraries and SDK. For information about these prerequisites, see IBM WebSphere MQ Telemetry system requirements.

Related information:

Migrating WebSphere MQ Telemetry from version 7.0.1 to version 7.5
Migrate IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5 by completing the tasks in this section. You must stop all IBM WebSphere MQ activity on the system before migrating.

WebSphere MQ Telemetry

Telemetry concepts and scenarios for monitoring and control

Administering WebSphere MQ Telemetry

Migration of telemetry applications from using WebSphere Message Broker version 6 to use WebSphere MQ Telemetry and WebSphere Message Broker version 7.0

Developing applications for WebSphere MQ Telemetry

WebSphere MQ Telemetry reference

Troubleshooting for WebSphere MQ Telemetry

Planning your installation on Windows systems

This topic describes the different methods available to install IBM WebSphere MQ on Windows systems and the different installation types .

If you are migrating from an earlier version of IBM WebSphere MQ, see Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 . To modify an existing installation, see “Modifying your installation” on page 244.

Interactive or Non-Interactive installation

IBM WebSphere MQ for Windows is installed using the Microsoft Installer (MSI). You can use the Installation Launchpad to invoke MSI, this process is called an attended or interactive installation. Or, you can invoke MSI directly for a silent installation, without using the IBM WebSphere MQ Installation Launchpad. This means that you can install IBM WebSphere MQ on a system without interaction. This process is called unattended, silent, or non-interactive installation, and is useful for installing IBM WebSphere MQ over a network on a remote system.

For a list of interactive and non-interactive features, see “IBM WebSphere MQ features for Windows systems” on page 173.

Interactive installation

If you choose an interactive installation, before you install, you must decide what type of installation you require. Table 32 on page 182 shows the installation types available, and the features that are installed with each option. For the prerequisites required for each feature, see IBM WebSphere MQ requirements .

The installation types are:

- Typical installation
- Compact installation
- Custom Installation

You can also:

- Specify the installation location, name, and description.
- Have multiple installations on the same computer.

See “Choosing a primary installation” on page 152 for important information about these features, including whether to designate your installation as the *primary installation*.

Table 32. Features installed with each type of interactive installation

Installation type	Server Features installed	Client Features installed	Comments
Typical	• Server • IBM WebSphere MQ Explorer • Development Toolkit • Java and .NET Messaging and Web Services	• Windows client • Development Toolkit • Java and .NET Messaging	The default option. Features are installed to default locations with a default installation name. Java and .NET Messaging and Web Services includes WebSphere MQ classes for .NET and support for the Microsoft Windows Communication Foundation (WCF) for use with Microsoft .NET 3.
Compact	• Server only	• Windows client only	The feature is installed to the default location with a default installation name.
Custom	By default, the following features are preselected: • Server • IBM WebSphere MQ Explorer • Development Toolkit • Java and .NET Messaging and Web Services A custom installation can also install: • Windows client • Client Extended Transaction Support • Java Extended Transaction Support • Telemetry (MQXR) Service • Telemetry Clients	By default, the following features are preselected: • Windows client • Development Toolkit • Java and .NET Messaging and Web Services A custom installation can also install: • Telemetry Clients	A server custom installation can be used if you want to install the Windows client from within the server image. All the available features are listed and you can select which ones to install, and where to install them. You can also name and provide a description for the installation. Use a custom installation when you want to specify that the installation is primary. Java and .NET Messaging and Web Services includes WebSphere MQ classes for .NET and support for the Microsoft Windows Communication Foundation (WCF) for use with Microsoft .NET 3 or later.

If Microsoft .NET is not installed before IBM WebSphere MQ and you add it, rerun **`setmqinst -i -n Installationname`** if this is a primary installation.

The following table describes which level of .NET is required for which function:

Table 33. Required levels of Microsoft .NET

WebSphere MQ function	.NET version required
WebSphere MQ classes for .NET. For more information, see: Getting started with WebSphere MQ classes for .NET 2	.NET 2
The WebSphere MQ custom channel for WCF. For more information, see: WebSphere MQ custom channel for WCF To build the sample solution files, either the Microsoft .NET 3.5 SDK, or Microsoft Visual Studio 2008 is needed. For more information, see: Software requirements for the WCF custom channel for WebSphere MQ	.NET 3.0 or later

For instructions on how to install IBM WebSphere MQ on Windows systems, see Installing WebSphere MQ Server on Windows systems and "Installing a IBM WebSphere MQ client on Windows systems" on page 265.

Non-interactive installation

If you choose a non-interactive installation the system on which you want to install must be able to access the IBM WebSphere MQ image, or a copy of the files, and you must be able to access the system.

If you are running IBM WebSphere MQ Version 7.5 or later, with User Account Control (UAC) enabled, you must invoke the non-interactive installation from an elevated command prompt. Elevate a command prompt by using a right-click to start the command prompt and choose **Run as administrator**. If you try to silently install from a non-elevated command prompt, the installation fails with an error of AMQ4353 in the installation log.

There are several ways to invoke MSI:

- Using the `msiexec` command with command-line parameters.
- Using the `msiexec` command with a parameter that specifies a response file. The response file contains the parameters that you normally supply during an interactive installation. See "Advanced installation using msiexec" on page 231.
- Use the `MQParms` command with command-line parameters, a parameter file, or both. The parameter file can contain many more parameters than a response file. See "Using the MQParms command" on page 239.

If the system belongs to a Windows domain you may need a special domain ID for the IBM WebSphere MQ service, see "Security considerations when installing WebSphere MQ server on a Windows system" for more information.

Related concepts:

"Security considerations when installing WebSphere MQ server on a Windows system"
Use this information to learn about the security considerations when installing IBM WebSphere MQ server on a Windows system.

Security considerations when installing WebSphere MQ server on a Windows system

Use this information to learn about the security considerations when installing IBM WebSphere MQ server on a Windows system.

- If you are installing IBM WebSphere MQ on a Windows domain network running Active Directory Server, you probably need to obtain a special domain account from your domain administrator. For further information, and the details that the domain administrator needs to set up this special account, see Configuring WebSphere MQ accounts.
- When you are installing IBM WebSphere MQ server on a Windows system you must have local administrator authority .
- In order to administer any queue manager on that system, or to run any of the IBM WebSphere MQ control commands your user ID must belong to the *local* `mqm` or `Administrators` group . If the local `mqm` group does not exist on the local system, it is created automatically when IBM WebSphere MQ is installed. A user ID can either belong to the local `mqm` group directly, or belong indirectly through the inclusion of global groups in the local `mqm` group.
- Windows versions with a User Account Control (UAC) feature restricts the actions users can perform on certain operating system facilities, even if they are members of the Administrators group. If your user ID is in the Administrators group but not the mqm group you must use an elevated command prompt to issue IBM WebSphere MQ admin commands such as crtmqm, otherwise the error AMQ7077 is generated. To open an elevated command prompt, right-click the start menu item, or icon, for the command prompt, and select **Run as administrator**
- Some commands can be run without being a member of the mqm group (see Authority to administer WebSphere MQ).
- If you intend to administer queue managers on a remote system, your user ID must be authorized on the target system.
- As with other versions of Windows, the object authority manager (OAM) gives members of the Administrators group the authority to access all IBM WebSphere MQ objects even when UAC is enabled.

Additional restrictions for installing on Windows

There are some additional points to consider when installing IBM WebSphere MQ Version 7.5 or later on Windows. First, Windows has some rules regarding the naming of objects created and used by IBM WebSphere MQ . Second, you can set up logging during installation which assists you in troubleshooting any problems you might have with the installation.

Naming considerations

- Ensure that the machine name does not contain any spaces. IBM WebSphere MQ does not support machine names that include spaces. If you install IBM WebSphere MQ on such a machine, you cannot create any queue managers.
- For IBM WebSphere MQ authorizations, names of user IDs and groups must be no longer than 64 characters (spaces are not allowed).
- A IBM WebSphere MQ for Windows server does not support the connection of a Windows client if the client is running under a user ID that contains the @ character, for example, abc@d. Similarly, the client user ID should not be the same as local group.
- A user account that is used to run the IBM IBM WebSphere MQ Windows service is set up by default during the installation process; the default user ID is MUSR_MQADMIN. This account is reserved for use by IBM WebSphere MQ. Refer to Configuring WebSphere MQ accounts.
- When a IBM WebSphere MQ client connects to a queue manager on the server, the username under which the client runs must not be same as the domain or machine name. If the user has the same name as the domain or machine, the connection fails with return code 2035(MQRC_NOT_AUTHORIZED).

Logging

Logging is enabled by default from the Launchpad. You can also enable complete logging, for more information, see How to enable Windows Installer logging

Planning your installation on HP Integrity NonStop Server

This section describes what to do to prepare your system for installing IBM WebSphere MQ client for HP Integrity NonStop Server.

Understanding multiple installations

IBM WebSphere MQ client for HP Integrity NonStop Server can be installed more than once on an HP Integrity NonStop Server system. In addition, multiple different versions of IBM WebSphere MQ can be installed on a single HP Integrity NonStop Server system, and be maintained independently. Each installation can be of any supported version of IBM WebSphere MQ. There are no requirements for installations to be either the same, or different versions.

To install IBM WebSphere MQ, you must specify two locations; one in the OSS file system, and one in the Guardian file system, which is used by the installer to store the results of the installation. These locations must not contain or overlap with any other IBM WebSphere MQ installation. The locations must also be free of other files.

Each installation is independent and self-contained, with all data, such as configuration logs, or trace and program files located within the installation directory hierarchy. All commands and libraries use an embedded runtime search path (RPath) to ensure that they load their dependencies from the same installation.

As several installations might be present, each application must locate and load the IBM WebSphere MQ client libraries from the correct installation.

- For native applications, an application that is linked with the IBM WebSphere MQ MQIC.LIB installation library inherits the IBM WebSphere MQ installation RPATH, and can run without environment variables. Environment variables in OSS, for example, *_RLD_LIB_PATH* or DEFINEs in Guardian, are only required if you want to run the application using a different IBM WebSphere MQ installation.
- For Java applications using the Java Messaging Service (JMS) API, the client Java archive (JAR) must be from the correct installation, and must be included on the class path. For more information, see Environment variables used by WebSphere MQ classes for JMS.

Product packaging and delivery

IBM WebSphere MQ client for HP Integrity NonStop Server is downloaded to the OSS file system as a single file.

The IBM WebSphere MQ client for HP Integrity NonStop Server package file is a self-extracting archive (SFX) that contains an installer and all files that are required to create installations.

The SFX for IBM WebSphere MQ client for HP Integrity NonStop Server has a file extension of `.run`. There is no concept of placed files. When run, the SFX creates a single installation, directly from the archive, into the OSS and Guardian file systems.

The SFX can be used to create as many installations of the IBM WebSphere MQ client for HP Integrity NonStop Server as you require. No information about installations is retained in the SFX, and no tools are provided for extracting individual files from the SFX.

File system

Before you install the IBM WebSphere MQ client for HP Integrity NonStop Server, make sure that the file system is set up correctly.

Review “Hardware and software requirements on HP Integrity NonStop Server systems” on page 194 to make sure that you understand the approximate amount of disk space in the OSS and Guardian file systems that is required for an installation. The OSS file set that is used for the installation requires enough free space for the installation files and the files you create in the installation. The Guardian volume that you use for installation does not require auditing.

Work with your systems administrator to verify the OSS file set and Guardian file system storage requirements, at least for an initial estimate of the storage. The best way to determine more precisely how much storage you would eventually need in production is to produce a prototype configuration and model the message storage requirements, scaling up as necessary for your production system.

OSS file system objects

For the OSS file system objects, this section concentrates on the differences between the HP Integrity NonStop Server installation, and the standard UNIX installation. Multiple independent installations are supported.

The opt and var trees must be present in a common root directory, which is selected at installation time. The opt tree contains files that do not change. For example, this tree contains program, library, dll, header files, and "static" data files. The var tree contains files that might change, and do hold status about the installation itself. Examples of files that this tree holds are configuration files, and log files.

Both the opt and var directories contain a single directory named mqm. The content of both trees is rooted in the opt/mqm and var/mqm directories.

This table shows a summary of the contents at the top level of opt/mqm:

Table 34.

Directory	Purpose	Contents
bin	Contains the OSS programs and libraries for an installation	• G is a symbolic link file that locates the Guardian installation subvolume • amq* files, containing the product executables for the client • lib* files, containing product dll files • Files containing control commands, and other utilities and scripts
inc	Contains the header files for building IBM WebSphere MQ applications	• .h files, which are C language header files • .tal files which are pTAL header files • .cpy files which are COBOL copy file • cobcpy32 and cobcpy64 directories for the individual COBOL copy files

Table 34. (continued)

Directory	Purpose	Contents
`lib`	Contains the import libraries needed to link applications	• `G` is a symbolic link file that locates the Guardian installation subvolume • `amq*` files, containing product dll files • `iconv` is a directory that contains data conversion tables • `lib*` files, which are product dll files • `mqicb` used for supplying to a CONSULT directive for compiling COBOL programs
`license`	Contains text versions of the IBM License for the IBM WebSphere MQ client for HP Integrity NonStop Server product, which is translated into each supported national language	• `Lic_.txt` files, which are the individual national language translations of the license. • `notices.txt` is a file that contains any additional license terms from non-IBM software that is included with IBM WebSphere MQ, if any
`mq.id`	Single file that contains information about the build level and install package	All contents in this directory might be used by IBM Support personnel.
`msg`	Contains globalization files for use by IBM WebSphere MQ, in logging and displaying output in the supported national language translations	The contents include: • `amq.cat` globalization message catalog currently in use by the installation, created by the OSS utility "gencat" • `amq.msg` unprocessed globalization data that is used as input by `gencat` to create the catalog • Other minor files and directories that support the different translations

Table 34. (continued)

Directory	Purpose	Contents
samp	Contains sample code and executables to illustrate the use of IBM WebSphere MQ	• *.cbl sample COBOL language source files • *.c sample 'C' language source files • *.tal sample pTAL language source files • ccsid.new backup file of ccsid.tbl • ccsid.tbl file that contains a table of supported CCSIDs • *.ini sample configuration files • java directory that contains source for sample Java applications • jms directory that contains source for sample JMS applications • bin directory that contains executable versions of the samples • dlq directory that contains a source for the sample Dead Letter Queue Handler • preconnect directory that contains source for preconnect exit

For more information about the samples that are provided with IBM WebSphere MQ client for HP Integrity NonStop Server, see Samples for IBM WebSphere MQ client for HP Integrity NonStop Server.

This table shows a summary of the contents at the top level of var/mqm:

Table 35.

Directory	Purpose	Contents
conv	Contains data conversion files	Binary data that supports the data conversion function for IBM WebSphere MQ
errors	Contains installation-wide error logs and FDC files	Standard content, for example: • AMQERR01.LOG - current system-wide error log file • AMQERR02.LOG - previous system-wide error log file • AMQERR03.LOG - oldest system-wide error log file • *.FDC FFST files
exits	Stores DLLs containing exit code that is loaded by the queue managers in the installation	This file is empty at installation
log	Contains log files for recording and controlling units of work	Standard content
mqs.ini	The installation configuration file	Standard content
qmgrs	Directory beneath the location where all of the queue manager directories are created	Standard content

Table 35. (continued)

Directory	Purpose	Contents
`sockets`	Directory tree that contains various queue manager control files	Standard content
`trace`	Defined location to which trace data is written by IBM WebSphere MQ	Standard content

Guardian installation subvolume

The Guardian single installation subvolume contains both the programs and libraries needed at runtime.

This table shows the contents of the Guardian installation subvolume:

Table 36.

File	Description
`AMQINST`	Internal file that describes installation configuration
`AMQS*`	Samples that are built for Guardian
`B*SAMP`	Sample build files for the various supported languages
`CMQ*`	Header files for the various supported languages, where files ending in: • `h` are C headers • `T` are pTAL headers • `L` are COBOL headers
`MQ*`	Product libraries
`MQS*C`	Sample C language source files
`MQS*T`	Sample pTAL language source files
`MQS*L`	Sample COBOL language source files

Control commands are also included, for a list, see HP Integrity NonStop Server client commands.

Checking requirements

Before you install IBM WebSphere MQ, you must check for the latest information and system requirements.

About this task

A summary of the tasks that you must complete to check system requirements are listed here with links to further information.

Procedure

1. Check that you have the latest information, including information on hardware and software requirements. See “Finding the latest information” on page 190.
2. Check that your systems meet the initial hardware and software requirements on your platform:
   - UNIX Linux “Hardware and software requirements on UNIX and Linux systems” on page 190
   - Windows “Hardware and software requirements on Windows systems” on page 192

The supported hardware and software environments are occasionally updated. See the IBM WebSphere MQ requirements website for the latest information.

3. Check that your systems have sufficient disk space for the installation. See Disk space requirements.
4. Check that you have the correct license requirements. See "License requirements" on page 195.

What to do next

When you have completed these tasks, you are ready to start preparing your systems for installation. For the next steps in installing IBM WebSphere MQ, see "Preparing the system" on page 196.

Related concepts:

"Installing IBM WebSphere MQ" on page 209
The topics in this section provide instructions on how to install IBM WebSphere MQ.

"Uninstalling" on page 304
The topics in this section provide instructions on how to uninstall components.

Related information:

WebSphere MQ maintenance tasks
This section describes how to apply and remove maintenance to IBM WebSphere MQ.

Finding the latest information

Access the latest information for IBM WebSphere MQ.

Requirements website
For the latest details of hardware and software requirements on all supported platforms, see the IBM WebSphere MQ requirements website.

readme.html
The product readme file is frequently updated and includes information about last minute changes and known problems and workarounds. The file is included on the product media and is installed when you install product components. The latest version is always on the product readmes web page. Always check to see that you have the latest copy.

Support information
The IBM WebSphere MQ support web page is regularly updated with the latest product support information. For example, if you are migrating from an earlier version, look under the heading *Solve a problem* for the document *Problems and solutions when migrating*.

Related concepts:

"Installing IBM WebSphere MQ" on page 209
The topics in this section provide instructions on how to install IBM WebSphere MQ.

Related information:

WebSphere MQ maintenance tasks
This section describes how to apply and remove maintenance to IBM WebSphere MQ.

Troubleshooting and support

Hardware and software requirements on UNIX and Linux systems

Before you install IBM WebSphere MQ, check that your system meets the hardware and operating system software requirements for the particular components you intend to install.

Hardware and software requirements are set out at the WebSphere MQ system requirements page.

IBM WebSphere MQ does not support host names that contain spaces. If you install IBM WebSphere MQ on a system with a host name that contains spaces, you are unable to create any queue managers.

Java Message Service and SOAP transport

If you want to use Java Message Service and SOAP support, you need an IBM Java 2 SDK and Runtime Environment Version 5.0 or later.

On Linux: Apache Axis V1.4 provides support for SOAP and is shipped on the server DVD, but not installed.

For a list of supported JDKs, see the IBM WebSphere MQ system requirements page at http://www.ibm.com/software/integration/wmq/requirements/.

For further information about using Java with IBM WebSphere MQ, see Using WebSphere MQ classes for Java.

For further information about SOAP with IBM WebSphere MQ, see WebSphere MQ transport for SOAP.

If you use a JDK that is not in the list of supported JDKs be aware that:

- The JDK might not be FIPS level 140-2 compliant. By using it with IBM WebSphere MQ, any SSL or TLS connections which use the unsupported Java runtime environment might not comply with FIPS 140-2 standards.
- SOAP is not supported.
- The IBM WebSphere MQ web service deployment utility, amqwdeployWMQService, requires IBM Java 2 SDK.

On HP-UX: To run a 64-bit or 32-bit JVM use the -d64 or -d32 parameters on the command line when running a Java application to ensure the correct JVM is used.

On Linux: On the Power® platform, the 32-bit and 64-bit JDKs are typically installed to different locations, for example, the 32-bit JDK is located in `/opt/IBMJava2-ppc-50` and the 64-bit JDK is located in `/opt/IBMJava2-ppc64-50`. Ensure that the PATH variable is correctly set for your applications that use Java. To use the Postcard application described in “Verify the installation using the Postcard application” on page 286, you must use a 32-bit JDK.

On Solaris: The 32-bit and 64-bit JDKs are typically installed to the same directory. To run a 64-bit JVM use the -d64 or -d32 parameters on the command line when running a Java application to ensure the correct JVM is used.

You can check the version installed using the following command:

```
java -version
```

Secure Sockets Layer (SSL)

If you want to use the SSL support, you need the IBM Global Security Kit (GSKit) V8 package. This package is supplied with IBM WebSphere MQ as one of the components available for installation.

HP-UX

To use SSL, IBM WebSphere MQ clients on HP-UX must be built using POSIX threads.

Linux

Installing the g++ version runtime support

If you intend to run SSL channels then you must have the g++ runtime libraries installed. The GNU g++ libraries are called `libgcc_s.so` and `libstdc++.so.6` and, on a Red Hat system with `libgcc` and `libstdc++` RPMs installed, are found in the directory `/usr/lib`. Check that the release level of the libraries meets IBM WebSphere MQ requirements, found at .

If you have a distribution that does not install libgcc_s.so and libstdc++.so.6 into /usr/lib, then you must do one of the following tasks:

- Locate and install the packages from your distribution vendor that contain these libraries.
- Install the GNU gcc and g++ compilers from another location, for example the home page for the GNU compiler collection at http://gcc.gnu.org/.

Ensure that the libraries listed are included in /usr/lib after installation.

The required packages for SSL support for IBM WebSphere MQ are set out on the following page: .

On 64 bit platforms, install both the 32 bit and the 64 bit versions of the package so that 32 bit and 64 bit processes can both use SSL functions.

IBM WebSphere MQ Explorer requirements

Linux If you want to use the IBM WebSphere MQ Explorer (available for use with IBM WebSphere MQ for Linux, (x86 and x86-64 platforms) only), your system requires the following things, as a minimum:

- 512 MB RAM
- 1 GHz processor
- 800 MB for Eclipse Platform code and data
- A suitable monitor for the operating system with a screen size of at least 1024x768
- Bitstream-vera-fonts

If you need to convert data to and from Unicode on your system, you must install the following file sets:

```
bos.iconv.ucs.com     Unicode converters for AIX sets
bos.iconv.ucs.ebcdic  Unicode converters for EBCDIC sets
bos.iconv.ucs.pc      Unicode converters for PC sets
```

Solaris 11 operating system

If you are installing on the Solaris 11 operating system, ensure that the IPS package (package/svr4) that supports **pkgadd** and equivalent utilities is installed.

Related concepts:

"Hardware and software requirements on Windows systems"
Check that the server environment meets the prerequisites for installing IBM WebSphere MQ for Windows and install any prerequisite software that is missing from your system from the server DVD.

Related tasks:

"Checking requirements" on page 189
Before you install IBM WebSphere MQ, you must check for the latest information and system requirements.

Hardware and software requirements on Windows systems

Check that the server environment meets the prerequisites for installing IBM WebSphere MQ for Windows and install any prerequisite software that is missing from your system from the server DVD.

Before installing IBM WebSphere MQ, you must check that your system meets the hardware and software requirements set out on the IBM WebSphere MQ system requirements page at http://www.ibm.com/software/integration/wmq/requirements/.

You must also review the release notes file, which is on the product DVD in the \Readmes folder for each national language, and check the READADD.txt file for any changes made between translation and the manufacturing of the installation DVD. `READADD.txt` is found in the root directory of the server installation DVD. During installation, the release notes file is copied to the IBM WebSphere MQ program files folder.

Storage requirements for IBM WebSphere MQ server

The storage requirements depend on which components you install, and how much working space you need. The storage requirements also depend on the number of queues that you use, the number and size of the messages on the queues, and whether the messages are persistent. You also require archiving capacity on disk, tape, or other media. For more information, see the IBM WebSphere MQ system requirements page at, http://www.ibm.com/software/integration/wmq/requirements/.

Disk storage is also required:

- Prerequisite software
- Optional software
- Your application programs

Requirements for IBM WebSphere MQ Explorer

If you want to use the IBM WebSphere MQ Explorer, the following minimum requirements apply:

- 512 MB of RAM
- A 1 GHz processor
- A monitor with a screen resolution of at least 1024 x 768 pixels

Install directory used for 64 bit Windows operating systems

On 64 bit Windows systems, IBM WebSphere MQ is installed into the default 32 bit installation location: `C:\Program Files (x86)\IBM\WebSphere MQ`.

If you select a non-default directory for installing Windows, do not try to install into `C:\Program Files\IBM\WebSphere MQ`. `C:\Program Files` can only contain 64 bit programs.

If you are doing a silent installation and you select `C:\Program Files\IBM\WebSphere MQ` as the installation directory, the installer writes an error to the error log and the installation fails.

Wherever the default installation location occurs in the documentation or `C:\Program Files\IBM\WebSphere MQ` is documented, for 64 bit operating systems the file path is `C:\Program Files (x86)\IBM\WebSphere MQ`.

Installing prerequisite software

To install the prerequisite software provided on the IBM WebSphere MQ Server DVD (which does not include service packs or web browsers), do one of the following:

- Use the IBM WebSphere MQ installation procedure.

  When you install using the IBM WebSphere MQ Server DVD, there is a **Software Prerequisites** option in the IBM WebSphere MQ Installation Launchpad window. You can use this option to check what prerequisite software is already installed and which is missing, and to install any missing software.
- Use Windows Explorer:
  1. Use Windows Explorer to select the `Prereqs` folder on the IBM WebSphere MQ Server DVD.
  2. Select the folder for the software item to be installed.
  3. Start the installation program.

Related concepts:

"Hardware and software requirements on UNIX and Linux systems" on page 190
Before you install IBM WebSphere MQ, check that your system meets the hardware and operating system software requirements for the particular components you intend to install.

Related tasks:

"Checking requirements" on page 189
Before you install IBM WebSphere MQ, you must check for the latest information and system requirements.

Hardware and software requirements on HP Integrity NonStop Server systems

Check that the server environment meets the prerequisites for installing the IBM WebSphere MQ client for HP Integrity NonStop Server. Check the product readme files and install missing prerequisite software supplied on the server CD.

Hardware

The IBM WebSphere MQ client for HP Integrity NonStop Server typically requires certain hardware specifications to run:

- HP Integrity NonStop Server H and J series
- Two or more processors
- At least 1 GB, and ideally 4 GB of memory per processor
- 500 MB of free disk space in the Guardian and OSS file systems

Operating system

Two operating systems are supported by the IBM WebSphere MQ client for HP Integrity NonStop Server:

- HP Integrity NonStop Server running H06.24 or later NonStop OS
- HP Integrity NonStop BladeSystem running J06.13 or later NonStop OS

You must be running one of these operating systems to install the IBM WebSphere MQ client for HP Integrity NonStop Server.

Other software requirements

IBM WebSphere MQ client for HP Integrity NonStop Server has some additional software requirements:

- The operating system software, Open System Services (OSS), must be active, with file systems and a local sockets subsystem that is configured and running.
- Safeguard must be active.
- If two-phase commit transaction support is required, then TMF must be active and Pathway must be configured and available. The connected queue manager must be at IBM WebSphere MQ Version 7.1 or later.
- If the Java Message Service (JMS) API is required, then HP Integrity NonStop Server for Java V6 must be available.
- You might require compatible compilers, linkers, and maybe other tools for the C, C++, COBOL, JMS, or pTAL languages if you want to build and use applications.

File system requirements

In the selected installation root directory, in the OSS file system, an installation creates:

- `opt` - a directory tree that contains the "static" files for an installation in OSS.

- `var` - a directory tree that contains the "variable" files for an installation in OSS.

An installation also creates a single subvolume in the Guardian file system, which is selected during installation.

Related concepts:

“Finding the latest information” on page 190
Access the latest information for IBM WebSphere MQ.

“License requirements”
You must have purchased sufficient licenses for your installation. The details of the license agreement is stored on your system at installation time so that you can read it at any time. IBM WebSphere MQ supports ITLM (IBM Tivoli® License Manager).

Related information:

UNIX Linux Disk space requirements
The storage requirements for WebSphere MQ depend on which components you install, and how much working space you need.

Verifying system software prerequisites

Use the HP Integrity NonStop Server TACL utility, SYSINFO, to verify the base OS level of the HP Integrity NonStop Server.

Procedure

From a TACL command prompt, enter **`SYSINFO`**.

Results

The system information is displayed as shown in the following example:

```
SYSINFO - T9268H01 - (01 OCT 2004)  SYSTEM \NODE1  Date 05 Nov 2010, 11:56:51
Copyright 2003 Hewlett-Packard Development Company, L.P.

         System name    \NODE1
  EXPAND node number    025
       Current SYSnn    SYS00
       System number    nnnnnn
 Software release ID    J06.10.00
```

In this example, the base OS level is J06.10.00.

What to do next

Compare the base OS level with the “Hardware and software requirements on HP Integrity NonStop Server systems” on page 194. Verify any other HP Integrity NonStop Server software prerequisites or recommendations identified in the documentation or the product README; for example, SPRs to particular products.

License requirements

You must have purchased sufficient licenses for your installation. The details of the license agreement is stored on your system at installation time so that you can read it at any time. IBM WebSphere MQ supports ITLM (IBM Tivoli License Manager).

License files

At installation, the license agreement files are copied into the `/licenses` directory under the *`MQ_INSTALLATION_PATH`*. You can read them at any time.

ITLM

If you are using ITLM, IBM WebSphere MQ automatically detects ITLM and checks with it each time a queue manager is started. You do not need to take any further action. You can install ITLM before or after IBM WebSphere MQ.

The automatic detection of ITLM applies to both the IBM WebSphere MQ server, and IBM WebSphere MQ Java products.

Related concepts:

UNIX Linux "Hardware and software requirements on UNIX and Linux systems" on page 190
Before you install IBM WebSphere MQ, check that your system meets the hardware and operating system software requirements for the particular components you intend to install.

Windows "Hardware and software requirements on Windows systems" on page 192
Check that the server environment meets the prerequisites for installing IBM WebSphere MQ for Windows and install any prerequisite software that is missing from your system from the server DVD.

Related tasks:

"Checking requirements" on page 189
Before you install IBM WebSphere MQ, you must check for the latest information and system requirements.

Preparing the system

On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

About this task

The tasks that you perform to prepare your systems for installation are listed here. Complete the appropriate tasks for your platform before installing.

Procedure

1. UNIX Linux On UNIX and Linux systems, set up the user and group. See "Setting up the user and group on UNIX and Linux systems" on page 197
2. UNIX Linux On UNIX and Linux, create file systems. See "Creating file systems on UNIX and Linux systems" on page 199
3. Configure additional settings for your platform:
   - AIX "Additional settings for installing on AIX systems" on page 201
   - HP-UX "Additional settings for installing on HP-UX systems" on page 202
   - Linux "Additional settings for IBM WebSphere MQ on Linux systems" on page 204
   - Solaris "Additional settings for installing on Solaris systems" on page 207
   - Windows "Additional restrictions for installing on Windows" on page 184

What to do next

When you have completed the tasks to prepare the system, you are ready to start installing IBM WebSphere MQ. To install a server, see "Installing a WebSphere MQ server" on page 209. To install a client, see "Installing a IBM WebSphere MQ client" on page 256.

Related information:

Planning
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Migrating and upgrading WebSphere MQ

WebSphere MQ maintenance tasks
This section describes how to apply and remove maintenance to IBM WebSphere MQ.

Setting up the user and group on UNIX and Linux systems

On UNIX and Linux systems, IBM WebSphere MQ requires a user ID of the name `mqm`, with a primary group of `mqm`. The `mqm` user ID owns the directories and files that contain the resources associated with the product.

Creating the user ID and group on UNIX and Linux systems

Set the primary group of the `mqm` user to the group `mqm`.

If you are installing IBM WebSphere MQ on multiple systems you might want to ensure each UID and GID of `mqm` has the same value on all systems. If you are planning to configure multi-instance queue managers, it is essential the UID and GID are the same from system to system. It is also important to have the same UID and GID values in virtualization scenarios.

AIX

You can use the System Management Interface Tool (**smit**), for which you require root authority.

1. To create the `mqm` group, display the required window using this sequence:

   ```
   Security & Users
     Groups
       Add a Group
   ```

   Set the group name field to `mqm`.
2. To create the user `mqm`, display the required window using this sequence:

   ```
   Security & Users
     Users
       Add a User
   ```

   Set the user name field to `mqm` .
3. To add a password to the new user ID, display the required window using this sequence:

   ```
   Security & Users
      Passwords
       Change a User's Password
   ```

   Set the password as required.

HP-UX

The user ID value for user `mqm` must be less than 60,000 to avoid problems with the maintenance update process.

You can use the System Management Homepage (SMH), or the **`groupadd`** and **`useradd`** commands to work with user IDs.

Linux RPM creates the `mqm` user ID and group ID as part of the installation procedure if they do not exist.

If you have special requirements for these IDs (for example they need to have the same values as other machines you are using, or your users and group ID are centrally managed) you should create the IDs before running the installation procedure, using the **groupadd** and **useradd** commands to set the UID and GID the same on each machine.

Note: The only IBM WebSphere MQ requirement, is that the mqm user should have the mqm group as its primary group.

Solaris

The user ID value for user mqm must be less than 262,143 to avoid problems with the maintenance update process.

Create the IDs using the **groupadd** and **useradd** commands to set the UID and GID the same on each machine.

Adding existing user IDs to the group on UNIX and Linux systems

If you want to run administration commands, for example **crtmqm** (create queue manager) or **strmqm** (start queue manager), your user ID must be a member of the mqm group. This user ID must not be longer than 12 characters.

Users do not need mqm group authority to run applications that use the queue manager; it is needed only for the administration commands.

AIX

You can use **smit** to add an existing user ID to the mqm group. Display the required menu using this sequence:

```
Security & Users
  Users
    Change / Show Characteristics of a User
```

Type the name of the user in the **User Name** field and press **Enter**. Add mqm to the **Group SET** field, which is a comma-separated list of the groups to which the user belongs. Users do not need to have their primary group set to mqm. If mqm is in their set of groups, they can use the administration commands.

Log files created by IBM WebSphere MQ Telemetry service

The **umask** setting of the user ID that creates a queue manager will determine the permissions of the Telemetry log files generated for that queue manager. Even though the ownership of the log files will be set to mqm .

Related concepts:

"Creating file systems on UNIX and Linux systems" on page 199
Before installing IBM WebSphere MQ Version 7.5, you might need to create file systems for both the product code and working data to be stored. There are minimum storage requirements for these file systems. The default installation directory for the product code can be changed at installation time, but the working data location cannot be changed.

"Additional settings for installing on AIX systems" on page 201

"Additional settings for installing on HP-UX systems" on page 202
Before you install IBM WebSphere MQ on an HP-UX system, you must check that the kernel is configured correctly.

"Additional settings for IBM WebSphere MQ on Linux systems" on page 204
Use this topic to when configuring IBM WebSphere MQ on Linux systems.

Related information:

"Additional settings for installing on Solaris systems" on page 207
Configure Solaris systems with the resource limits required by IBM WebSphere MQ.

Setting up the user and group on HP Integrity NonStop Server

The administrator user ID must be used to administer the IBM WebSphere MQ client for HP Integrity NonStop Server.

Ensure that you have access to a IBM WebSphere MQ client for HP Integrity NonStop Server user ID in the user group called MQM. The MQM group must be created before the client can be installed. All user IDs that are used to install the client must have MQM as their primary group. If this user group does not exist, or you do not have access to such a user, contact your systems administrator.

Creating file systems on UNIX and Linux systems

Before installing IBM WebSphere MQ Version 7.5, you might need to create file systems for both the product code and working data to be stored. There are minimum storage requirements for these file systems. The default installation directory for the product code can be changed at installation time, but the working data location cannot be changed.

Determining the size of a server installations file system

To determine the size of the `/var/mqm` file system for a server installation, consider:

- The maximum number of messages in the system at one time.
- Contingency for message buildups, if there is a system problem.
- The average size of the message data, plus 500 bytes for the message header.
- The number of queues.
- The size of log files and error messages.
- The amount of trace that is written to the `/var/mqm/trace` directory.

Storage requirements for IBM WebSphere MQ also depend on which components you install, and how much working space you need. For more details, see Disk space requirements.

Creating a file system for the working data

Before you install IBM WebSphere MQ, create and mount a file system called `/var/mqm` which is owned by the user `mqm` in the group `mqm`; see “Setting up the user and group on UNIX and Linux systems” on page 197. This file system is used by all installations of IBM WebSphere MQ on a system. If possible, use a partition strategy with a separate volume for the IBM WebSphere MQ data. This means that other system activity is not affected if a large amount of IBM WebSphere MQ work builds up. Configure the directory permissions to permit the `mqm` user to have full control, for example, file mode 755. These permissions will then be updated during the WebSphere MQ installation to match the permissions required by the queue manager.

Creating separate file systems for errors and logs

You can also create separate file systems for your log data (`/var/mqm/log`) and error files (`/var/mqm/errors`). If possible, place these directories on different physical disks from the queue manager data (`/var/mqm/qmgrs`) and from each other.

If you create separate file systems the `/var/mqm/errors` directory can be NFS mounted. However, if you choose to NFS-mount `/var/mqm/errors`, the error logs might be lost if the network fails.

You can protect the stability of your queue manager by having separate file systems for:

- `/var/mqm/errors`
- `/var/mqm/trace`
- `/var/mqm/qmgrs`
- `/var/mqm/log`

In the case of /var/mqm/errors, it is rare that this directory receives large quantities of data. But it is sometimes seen, particularly if there is a severe system problem leading to IBM WebSphere MQ writing a lot of diagnostic information in to .FDC files. In the case of /var/mqm/trace, files are only written here when you use **strmqtrc** to start tracing IBM WebSphere MQ.

You can obtain better performance of normal IBM WebSphere MQ operations (for example, syncpoints, MQPUT, MQGET of persistent messages) by placing the following on separate disks:

- /var/mqm/qmgrs
- /var/mqm/log

In the rare event that you need to trace a IBM WebSphere MQ system for problem determination, you can reduce performance impact by placing the /var/mqm/trace file system on a separate disk.

If you are creating separate file systems, allow a minimum of 30 MB of storage for /var/mqm, 100 MB of storage for /var/mqm/log, and 10 MB of storage for /var/mqm/errors. The 100 MB minimum allowance of storage for /var/mqm/log is the absolute minimum required for a single queue manager and is not a recommended value. The size of a file system must be scaled according to the number of queue managers that you intend to use, the number of pages per log file, and the number of log files per queue manager.

If you want to use individual queues that hold more than 2 GB of data, you must enable /var/mqm to use large files.

For more information about file systems, see File system support.

The size of the log file depends on the log settings that you use. The minimum sizes are for circular logging using the default settings. For more information about log sizes, see Calculating the size of the log.

Linux and Solaris

For a client installation, the file system can be mounted on a remote network device, for example NFS.

If you are performing both a client and a server installation, the requirements of the server installation take precedence over the requirements of the client installation.

Allow 15 MB as a minimum for a IBM WebSphere MQ client.

A new sample IBM WebSphere MQ MQI client configuration file is created in the var/mqm directory, by the client package, during installation, but only if this file does not exist. This file contains the ClientExitPath stanza. An example mqclient.ini file is shown in Configuring a client using a configuration file WebSphere MQ MQI client configuration file.

If you are using a common configuration file for multiple clients, either in the IBM WebSphere MQ installation directory or in another location using the MQCLNTCF environment variable, you must grant read access to all user identifiers under which the IBM WebSphere MQ client applications run. If, for any reason, the file cannot be read, the failure is traced, and the search logic continues as if the file had not existed.

Related concepts:

"Setting up the user and group on UNIX and Linux systems" on page 197
On UNIX and Linux systems, IBM WebSphere MQ requires a user ID of the name mqm, with a primary group of mqm. The mqm user ID owns the directories and files that contain the resources associated with the product.

"Additional settings for installing on AIX systems"

"Additional settings for installing on HP-UX systems" on page 202
Before you install IBM WebSphere MQ on an HP-UX system, you must check that the kernel is configured correctly.

"Additional settings for IBM WebSphere MQ on Linux systems" on page 204
Use this topic to when configuring IBM WebSphere MQ on Linux systems.

Related tasks:

"Preparing the system" on page 196
On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

Related information:

"Additional settings for installing on Solaris systems" on page 207
Configure Solaris systems with the resource limits required by IBM WebSphere MQ.

Additional settings for installing on UNIX and Linux systems

Some UNIX and Linux and Linux systems require you to make additional settings.

- "Additional settings for installing on AIX systems"
- "Additional settings for installing on HP-UX systems" on page 202
- "Additional settings for IBM WebSphere MQ on Linux systems" on page 204
- "Additional settings for installing on Solaris systems" on page 207

Additional settings for installing on AIX systems

File descriptors

When running a multi-threaded process such as the agent process, you might reach the soft limit for file descriptors. This limit gives you the IBM WebSphere MQ reason code MQRC_UNEXPECTED_ERROR (2195) and, if there are enough file descriptors, a IBM WebSphere MQ FFST file.

To avoid this problem, increase the process limit for the number of file descriptors. You must alter the nofiles attribute in /etc/security/limits to 10,000 for the mqm user ID, or in the default stanza. To alter the number of file descriptors do these steps:

1. In a command prompt, check the maximum number of file descriptors available to a process running as mqm:

   ```
   lsuser -a nofiles mqm
   ```

2. Set the value to at least 10240:

   ```
   chuser nofiles_hard=10240 mqm
   chuser nofiles=10240 mqm
   ```

System Resource Limits

Set the system resource limit for data segment and stack segment to unlimited using the following commands in a command prompt:

```
ulimit -d unlimited
ulimit -s unlimited
```

For more information on configuring your system, see How to configure UNIX and Linux systems for WebSphere MQ.

Related concepts:

“Setting up the user and group on UNIX and Linux systems” on page 197
On UNIX and Linux systems, IBM WebSphere MQ requires a user ID of the name mqm, with a primary group of mqm. The mqm user ID owns the directories and files that contain the resources associated with the product.

“Creating file systems on UNIX and Linux systems” on page 199
Before installing IBM WebSphere MQ Version 7.5, you might need to create file systems for both the product code and working data to be stored. There are minimum storage requirements for these file systems. The default installation directory for the product code can be changed at installation time, but the working data location cannot be changed.

Related tasks:

“Preparing the system” on page 196
On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

Additional settings for installing on HP-UX systems

Before you install IBM WebSphere MQ on an HP-UX system, you must check that the kernel is configured correctly.

Kernel configuration

It is possible that the default kernel configuration is not adequate because IBM WebSphere MQ uses semaphores and shared memory.

Before installation, review the configuration of the machine and increase the values if necessary. Consider using the values of the tunable kernel parameters given in Figure 62. These values might need to be increased if you obtain any First Failure Support Technology™ (FFST) records.

Note:

1. Semaphore and swap usage do not vary significantly with message rate or message persistence.
2. WebSphere MQ queue managers are independent of each other. Therefore system tunable kernel parameters, for example shmmni, semmni, semmns, and semmnu need to allow for the number of queue managers in the system.

See the HP-UX documentation for information about changing these values.

```
shmmax          536870912
shmseg          1024
shmmni          1024
semaem          16384
semvmx          32767
semmns          16384
semmni          1024 (semmni < semmns)
semmnu          16384
semume          256
max_thread_proc  66
maxfiles        10000
maxfiles_lim    10000
```

Figure 62. Minimum tunable kernel parameters values

You must restart the system after you change any of the tunable kernel parameters.

System resource limits

You can set global limits for the size of process data segments and the size of process stack segments for the whole system. These limits are set by altering the tunable kernel parameters.

The tunable kernel parameters are:

Parameter	What it controls	Consider minimum value
maxdsiz	Maximum size of the data segment for 32-bit processes	1073741824
maxdsiz_64bit	Maximum size of the data segment for 64-bit processes	1073741824
maxssiz	Maximum size of the stack segment for 32-bit processes	8388608
maxssiz_64bit	Maximum size of the stack segment for 64-bit processes	8388608

If other software on the same machine needs higher values, then the operation of IBM WebSphere MQ is not adversely affected if those higher values are used.

For the full documentation for these parameters see the HP-UX product documentation.

To apply the settings to an HP-UX 11i system which has the System Administration Manager (SAM) utility, you can use SAM to achieve the following steps:

- Select and alter the parameters
- Process the new kernel
- Apply the changes and restart the system

Other releases of HP-UX might provide different facilities to set the tunable kernel parameters. Consult your HP-UX product documentation for the relevant information.

The `ulimit` shell command

On a per-shell basis, the available limits can be tuned down from the values stored for the "System resource limits" preceding parameters. Use the **ulimit** shell command to tune the values of the parameters with a combination of the following switches:

Switch	Meaning
-H	The hard limit
-S	The soft limit
-d	The data segments size
-s	The stack segment size

Verifying that the kernel settings are applied

You can verify that the resource limits have not been lowered by a **ulimit** command and that the queue manager experiences the correct limits. To do so, go to the shell from which the queue manager is started and enter the following command:

```
ulimit -Ha
ulimit -Sa
```

Among the console output you can see:

```
data(kbytes)   1048576
stack(kbytes)  8192
```

If lower numbers are returned, then a **ulimit** command has been issued in the current shell to lower the limits. Consult with your system administrator to resolve the issue.

For more information on configuring your system, see How to configure UNIX and Linux systems for WebSphere MQ.

Related concepts:

"Setting up the user and group on UNIX and Linux systems" on page 197
On UNIX and Linux systems, IBM WebSphere MQ requires a user ID of the name mqm, with a primary group of mqm. The mqm user ID owns the directories and files that contain the resources associated with the product.

"Creating file systems on UNIX and Linux systems" on page 199
Before installing IBM WebSphere MQ Version 7.5, you might need to create file systems for both the product code and working data to be stored. There are minimum storage requirements for these file systems. The default installation directory for the product code can be changed at installation time, but the working data location cannot be changed.

Related tasks:

"Preparing the system" on page 196
On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

Additional settings for IBM WebSphere MQ on Linux systems

Use this topic to when configuring IBM WebSphere MQ on Linux systems.

Installation DVD

Ensure that you have the correct 32-bit or 64-bit server DVD. If you want to install the 64-bit Server, you must mount the 64-bit Server DVD.

Shell interpreter

Ensure that /bin/sh shell is a valid shell interpreter compatible with the Bourne shell, otherwise the post-installation configuration of IBM WebSphere MQ does not complete successfully. If the shell was not installed by using RPM, you might see a prerequisites failure of /bin/sh shell when you try to install IBM WebSphere MQ. The failure is because the RPM tables do not recognize that a valid shell interpreter is installed. If the failure occurs, you can reinstall the /bin/sh shell by using RPM, or specify the RPM option --nodeps to disable dependency checking during installation of IBM WebSphere MQ.

Note: The **--dbpath** option is not supported when installing IBM WebSphere MQ on Linux.

Kernel configuration

IBM WebSphere MQ uses System V IPC resources, in particular shared memory and semaphores.

The minimum configuration for IBM WebSphere MQ for these resources is as follows:

```
kernel.shmmni = 4096
kernel.shmall = 2097152
kernel.shmmax = 268435456
kernel.sem = 500 256000 250 1024
fs.file-max = 524288
kernel.pid-max = 12000
```

```
   kernel.threads-max = 48000
```

If you plan to run more than one queue manager of moderate size on the server, increase the file-max parameter, **fs.file-max**.

To view the kernel parameters for your system, enter the following commands:

```
   cat /proc/sys/kernel/shmmni
   cat /proc/sys/kernel/shmall
   cat /proc/sys/kernel/shmmax
   cat /proc/sys/kernel/sem
   cat /proc/sys/fs/file-max
```

Each of these commands returns the value of the corresponding kernel parameter. For example, `cat /proc/sys/kernel/shmmni` returns the value for *kernel.shmmni*. If any of the values is less than the minimum value, you need to increase it to at least the minimum value.

To add or alter these values, log on as a user with root authority. Open the file `/etc/sysctl.conf` with a text editor, then add or change the following entries to the values shown:

```
kernel.shmmni = 4096
kernel.shmall = 2097152
kernel.shmmax = 268435456
kernel.sem = 500 256000 250 1024
fs.file-max = 524288
```

Then save and close the file.

To load these **sysctl** values immediately, enter the following command:

```
sysctl -p
```

If you do not issue the `sysctl -p` command, the new values are loaded when the system is rebooted.

By default the Linux kernel has a maximum process identifier, that can also be used with threads, and might limit the allowed number of threads.

The operating system reports when the system lacks the necessary resources to create another thread, or the system-imposed limit on the total number of threads in a process {PTHREAD_THREADS_MAX} would be exceeded.

For more information on `kernel.threads-max` and `kernel.pid-max`, see Resource shortage in WebSphere MQ queue manager when running a large number of clients

Maximum open files

If the system is heavily loaded, you might need to increase the maximum possible number of open files. If your distribution supports the `proc` file system you can query the current limit by issuing the following command:

```
cat /proc/sys/fs/file-max
```

.

To report on the current maximum, and in-use, number of file descriptors for your system, enter the following commands:

```
   /sbin/sysctl fs.file-max
   /sbin/sysctl fs.file-nr
```

If you are using a pluggable security module such as PAM (Pluggable Authentication Module), ensure that this module does not unduly restrict the number of open files for the mqm user. To report the maximum number of open file descriptors per process for the mqm user, login as the mqm user and enter the following values:

```
ulimit -n
```

For a standard IBM WebSphere MQ queue manager, set the *nofile* value for the mqm user to 10240 or more. To set the maximum number of open file descriptors for processes running under the mqm user, add the following information to the /etc/security/limits.conf file:

```
mqm           hard   nofile        10240
mqm           soft   nofile        10240
```

Maximum processes

A running IBM WebSphere MQ queue manager consists of a number of thread programs. Each connected application increases the number of threads running in the queue manager processes. It is normal for an operating system to limit the maximum number of processes that a user runs. The limit prevents operating system failures due to an individual user or subsystem creating too many processes. You must ensure that the maximum number of processes that the mqm user is allowed to run is sufficient. The number of processes must include the number of channels and applications that connect to the queue manager.

The following calculation is useful when determining the number of processes for the mqm user:

```
nproc = 2048 + clientConnections * 4 + qmgrChannels * 4 +
    localBindingConnections
```

where:
- *clientConnections* is the maximum number of connections from clients on other machines connecting to queue managers on this machine.
- *qmgrChannels* is the maximum number of running channels (as opposed to channel definitions) to other queue managers. This includes cluster channels, sender/receiver channels, and so on.
- *localBindingConnections* does not include application threads.

The following assumptions are made in this algorithm:
- 2048 is a large enough contingency to cover the queue manager threads. This might need to be increased if a lot of other applications are running.
- When settting nproc, take into account the maximum number of applications, connections, channels and queue managers that might be run on the machine in the future.
- This algorithm takes a pessimistic view and the actual nproc needed might be slightly lower for later versions of IBM WebSphere MQ and fastpath channels.
- V7.5.0.8 In Linux, each thread is implemented as a light-weight process (LWP) and each LWP is counted as one process against nproc.

You can use the PAM_limits security module to control the number of processes that users run. You can configure the maximum number of processes for the mqm user as follows:

```
mqm           hard   nproc         4096
mqm           soft   nproc         4096
```

For more details on how to configure the PAM_limits security module type, enter the following command:

```
man limits.conf
```

For more information on configuring your system, see How to configure UNIX and Linux systems for WebSphere MQ.

Related concepts:

"Setting up the user and group on UNIX and Linux systems" on page 197
On UNIX and Linux systems, IBM WebSphere MQ requires a user ID of the name mqm, with a primary group of mqm. The mqm user ID owns the directories and files that contain the resources associated with the product.

"Creating file systems on UNIX and Linux systems" on page 199
Before installing IBM WebSphere MQ Version 7.5, you might need to create file systems for both the product code and working data to be stored. There are minimum storage requirements for these file systems. The default installation directory for the product code can be changed at installation time, but the working data location cannot be changed.

Related tasks:

"Preparing the system" on page 196
On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

Additional settings for installing on Solaris systems

Configure Solaris systems with the resource limits required by IBM WebSphere MQ.

IBM WebSphere MQ uses semaphores, shared memory, and file descriptors, and it is probable that the default resource limits are not adequate.

For further information on **maxusers**, and other process-sizing parameters, on Solaris 10, see Process sizing parameters.

If you are using Solaris 10, or later versions, you must change the default resource limits for each zone IBM WebSphere MQ will be installed in. To set new default limits for all users in the *mqm* group, set up a project for the *mqm* group in each zone.

To find out if you already have a project for the *mqm* group, log in as root and enter the following command:

```
projects -l
```

If you do not already have a *group.mqm* project defined, enter the following command:

```
projadd -c "WebSphere MQ default settings"
        -K "process.max-file-descriptor=(basic,10000,deny)"
        -K "project.max-shm-memory=(priv,4GB,deny)"
        -K "project.max-shm-ids=(priv,1024,deny)"
        -K "project.max-sem-ids=(priv,1024,deny)" group.mqm
```

If a project called *group.mqm* is listed, review the attributes for that project. The attributes must include the following minimum values:

```
process.max-file-descriptor=(basic,10000,deny)
project.max-sem-ids=(priv,1024,deny)
project.max-shm-ids=(priv,1024,deny)
project.max-shm-memory=(priv,4294967296,deny)
```

If you need to change any of these values, enter the following command:

```
projmod -s -K "process.max-file-descriptor=(basic,10000,deny)"
           -K "project.max-shm-memory=(priv,4GB,deny)"
           -K "project.max-shm-ids=(priv,1024,deny)"
           -K "project.max-sem-ids=(priv,1024,deny)" group.mqm
```

Note that you can omit any attributes from this command that are already correct.

For example, to change only the number of file descriptors, enter the following command:

```
projmod -s -K "process.max-file-descriptor=(basic,10000,deny)" group.mqm
```

(To set only the limits for starting the queue manager under the mqm user, login as mqm and enter the command `projects` . The first listed project is likely to be `default` , and so you can use `default` instead of `group.mqm`, with the `projmod` command.)

You can find out what the file descriptor limits for the current project are, by compiling and running the following program.

Before running the program you must create the `tryfd` file in the current directory:

```
/*********************************************************
You must make the tryfd file in the current directory first
before running this program..
*********************************************************/
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

main() {
int b=0;
int fd;
  for (;;) {
    fd = open ("./tryfd", O_RDONLY);
    if (fd == -1) {
        printf ("fd is %d\n", b);
        perror("fd");
          break;
        }
        b++;
  }
}
```

To ensure that the attributes for the project `group.mqm` are used by a user session when running Websphere MQ, make sure that the primary group of that user ID is mqm. In the examples in this topic, the `group.mqm` project ID will be used. For further information on how projects are associated with user sessions, see Sun's *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones* for your release of Solaris.

For more information on configuring your system, see How to configure UNIX and Linux systems for WebSphere MQ.

Related concepts:

"Setting up the user and group on UNIX and Linux systems" on page 197
On UNIX and Linux systems, IBM WebSphere MQ requires a user ID of the name `mqm`, with a primary group of `mqm`. The `mqm` user ID owns the directories and files that contain the resources associated with the product.

"Creating file systems on UNIX and Linux systems" on page 199
Before installing IBM WebSphere MQ Version 7.5, you might need to create file systems for both the product code and working data to be stored. There are minimum storage requirements for these file systems. The default installation directory for the product code can be changed at installation time, but the working data location cannot be changed.

Related tasks:

"Preparing the system" on page 196
On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

Installing IBM WebSphere MQ

The topics in this section provide instructions on how to install IBM WebSphere MQ.

See “Finding the latest information” on page 190 for details about how to check that you have access to the most recent information available.

If product fixes or updates are made available, see IBM WebSphere MQ maintenance tasks for information about how to apply these changes.

To prepare for installation and to install the IBM WebSphere MQ components, complete the following tasks:
- “Planning your installation” on page 149
- “Checking requirements” on page 189
- “Preparing the system” on page 196
- “Installing a WebSphere MQ server”
- “Installing a IBM WebSphere MQ client” on page 256
- “Installing IBM WebSphere MQ Telemetry” on page 179

Related concepts:

“Verifying a server installation” on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

“Verifying a client installation” on page 290
You can verify that your WebSphere MQ MQI client installation completed successfully and that the communication link is working.

“Multiple installations” on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

Related tasks:

Installing WebSphere MQ Advanced Message Security
Install and uninstall IBM WebSphere MQ Advanced Message Security component.

Related information:

Migrating and upgrading WebSphere MQ

WebSphere MQ Managed File Transfer product options

Installing a WebSphere MQ server

After preparing your system for installation you may install IBM WebSphere MQ by following the appropriate instructions for your platform. After installation, you might want to verify your installation to check that installation has been successful.

Before you begin

Make sure that you have prepared your system. See Preparing the system.

About this task

It is possible to have both a server and a client installation on the same machine, for instructions on how to do this see, “Installing a IBM WebSphere MQ client” on page 256.

IBM WebSphere MQ Telemetry is installed as part of the IBM WebSphere MQ Server installation. It must be selected as part of a custom installation. For more information, see “Installing IBM WebSphere MQ Telemetry” on page 179

Procedure

To begin the installation procedure, select the appropriate platform and installation method.

- Interactive installation
  1. “Installing IBM WebSphere MQ server on AIX”
  2. “Installing IBM WebSphere MQ server on HP-UX” on page 214
  3. “Installing IBM WebSphere MQ server on Linux” on page 218
  4. “Installing IBM WebSphere MQ server on Linux Ubuntu” on page 220
  5. “Installing IBM WebSphere MQ server on Solaris” on page 224
  6. “Installing IBM WebSphere MQ server on Windows” on page 229
- Non-interactive installation
  1. “Non-interactive installation of IBM WebSphere MQ server on AIX” on page 213
  2. “Non-interactive installation of IBM WebSphere MQ server on HP-UX” on page 216
  3. “Non-interactive installation of IBM WebSphere MQ server on Solaris” on page 227
  4. Non-interactive installation of WebSphere MQ server on Windows systems

Related concepts:

“Planning your installation” on page 149
Before you install IBM WebSphere MQ, you must choose which components to install and where to install them. You must also make some platform-specific choices.

“Installing a IBM WebSphere MQ client” on page 256

“Installing IBM WebSphere MQ Telemetry” on page 179
From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

Related tasks:

“Displaying messages in your national language on UNIX and Linux systems” on page 255
To display messages from a different national language message catalog, you must install the appropriate catalog and set the **LANG** environment variable.

“Checking requirements” on page 189
Before you install IBM WebSphere MQ, you must check for the latest information and system requirements.

“Preparing the system” on page 196
On some operating systems, you might have to complete several tasks before you install IBM WebSphere MQ depending on your installation platform. You might also want to complete other tasks, depending on your installation intentions.

“Converting a trial license on UNIX, Linux, and Windows” on page 254
Convert a trial license to a full license without reinstalling IBM WebSphere MQ.

Installing IBM WebSphere MQ server on AIX

You can install IBM WebSphere MQ server on an AIX system.

Before you begin

- Before you start the installation procedure, make sure that you have completed the necessary steps outlined in “Preparing the system” on page 196.
- WebSphere MQ can be installed into System Workload Partitions (WPARs) with both shared and private file systems. For installation into private file systems, WebSphere MQ can be installed directly

into the System WPAR using the procedure outlined in this topic. For installation into shared file systems, see Installing WebSphere MQ in AIX Workload Partitions. There are some limitations for shared `/usr` file systems:

  - The **dspmqinst** and **dspmqver** commands might report the primary installation incorrectly when compared with the symbolic links in `/usr/bin`. To synchronize the reporting of the primary installation in a System WPAR and the global environment, run **setmqinst** with the **-i** or **-x** parameter, on the individual zones.
  - You cannot change the primary installation within a WPAR. You must change the primary installation through the global environment which has appropriate write access to `/usr/bin`.

  Note: During installation to a non-default location, ATTENTION messages that relate to **errupdate** or **trcupdate** are produced. These messages are not errors. However, AIX system trace for IBM WebSphere MQ is not supported for installations in a non-default location, and IBM WebSphere MQ trace must be used for problem determination.

- If you install a copy of IBM WebSphere MQ server for AIX obtained from Passport Advantage®, you need to:
  - Uncompress the tar file, by using the following command:

    ```
    uncompress  WS_MQ_V7.5_TRIAL_FOR_AIX_ML.tar.z
    ```
  - Extract the installation files from the tar file, by using the following command:

    ```
    tar -xvf  WS_MQ_V7.5_TRIAL_FOR_AIX_ML.tar
    ```
  - Use the installation tools, **installp** or **smit**, to install the IBM WebSphere MQ server for AIX.

About this task

IBM WebSphere MQ is supplied as a set of filesets that are installed using the standard AIX installation tools. The procedure uses the system management interface tool (SMIT), but you can choose to use **installp**, **geninstall** or the web-based System Manager. You can select which components you want to install. The components and file sets are listed in “Choosing what to install” on page 159.

This procedure installs IBM WebSphere MQ into the default location of `/usr/mqm`.

If you want to install IBM WebSphere MQ in any one of the following situations:

- As the first installation on your system using **installp**
- As the first installation on your system, and you are installing the product to a location that is not the default
- Alongside an existing installation

use the procedure described in “Non-interactive installation of IBM WebSphere MQ server on AIX” on page 213.

If you want to carry out a side-by-side installation, alongside an existing installation of IBM WebSphere MQ in the default location, the existing installation must be IBM WebSphere MQ Version 7.0.1.6, or later.

You must install the second version of the product in a location that is not the default, using **installp** (see “Non-interactive installation of IBM WebSphere MQ server on AIX” on page 213).

If you want to carry out a single stage migration, refer to one of the following topics:

- “AIX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 419
- “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Set your current directory to the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. List the software in the **SOFTWARE to install** field:
   a. Enter **.**
   b. Press **F4**
4. Select the required **smit** window using the following sequence:

```
Software Installation and Maintenance
 Install and Update Software
  Install and Update from ALL Available Software
```

5. Select the filesets to install from the list. Ensure that you include the appropriate message catalog if you require messages in a language different from the language specified by the locale selected on your system. Enter **ALL** to install all applicable filesets.
6. View the license agreement:
   a. Change **Preview new LICENSE agreements?** to **yes**
   b. Press **Enter**
7. Accept the license agreements and install IBM WebSphere MQ:
   a. Change **ACCEPT new license agreements?** to **yes**
   b. Change **Preview new LICENSE agreements?** to **no**
   c. Press **Enter**

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  where *MQ_INSTALLATION_PATH* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Choosing an installation location" on page 157
You can install IBM WebSphere MQ to a custom location during the installation process. Alternatively, you can install to the default location. The location where IBM WebSphere MQ is installed is known as the *MQ_INSTALLATION_PATH*.

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Non-interactive installation of IBM WebSphere MQ server on AIX" on page 213
Install IBM WebSphere MQ server from the command line using the AIX **installp** command.

"Uninstalling IBM WebSphere MQ on AIX" on page 304
On AIX, you can uninstall theIBM WebSphere MQ server or client using the System Management Interface Tool (SMIT) or the **installp** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Non-interactive installation of IBM WebSphere MQ server on AIX:

Install IBM WebSphere MQ server from the command line using the AIX **installp** command.

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

Note: During installation, errors relating to **errupdate** or **trcupdate** might occur. This can caused by installing to a non-default location, if so these errors can be safely ignored. However, native trace for IBM WebSphere MQ is only supported when installed in the default location.

About this task

You can use this method to install to a non-default location, and can select which components you want to install. The components and filesets are listed in "Choosing what to install" on page 159.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Set your current directory to the location of the installation file. The location might be the mount point of the CD, a network location, or a local file system directory.
3. Install the product in one of the following ways:
   - Install the whole product in the default location:
     ```
     installp -acgXYd . all
     ```
   - Install selected filesets in the default location:
     ```
     installp -acgXYd . list of file sets
     ```
   - Install the whole product in a non-default location using the -R flag:
     ```
     installp -R USIL_Directory -acgXYd . all
     ```
   - Install selected filesets in a non-default location using the -R flag:
     ```
     installp -R USIL_Directory -acgXYd . list of file sets
     ```

   where *USIL_Directory* is a directory which exists before the command is run; it must not contain any spaces or usr/mqm. IBM WebSphere MQ is installed underneath the directory specified. For example, if /USIL1 is specified, the IBM WebSphere MQ product files are located in /USIL1/usr/mqm. This location is known as the *MQ_INSTALLATION_PATH*.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:
  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```
  where *MQ_INSTALLATION_PATH* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .

- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Installing IBM WebSphere MQ server on AIX" on page 210
You can install IBM WebSphere MQ server on an AIX system.

"Uninstalling IBM WebSphere MQ on AIX" on page 304
On AIX, you can uninstall theIBM WebSphere MQ server or client using the System Management Interface Tool (SMIT) or the **installp** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

User Specified Installation Location (USIL)

Installing IBM WebSphere MQ server on HP-UX

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

About this task

This task describes the installation of a server, using the **swinstall** program to select which components you want to install. The components are listed in "Choosing what to install" on page 159.

Note: If you are using a screen reader, use the non-interactive installation option "Non-interactive installation of IBM WebSphere MQ server on HP-UX" on page 216, so that you can accept the license without viewing it.

If you are installing IBM WebSphere MQ from a depot that contains service update packages, read HP-UX: Applying maintenance level updates on IBM WebSphere MQ Version 7.5 before installing the service update packages.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Set your current directory to the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. Accept the license by running the `mqlicense` script:
   ```
   ./mqlicense.sh
   ```
   The license is displayed. If you accept the license, you can continue the installation.
4. Start the interactive installation procedure by typing the following command,

```
swinstall -s /installation_file
```

/installation_file is the absolute path to the installation file. The path must begin with a / and end with the name of the installation file. In the resulting menu screen, select **MQSERIES**.

   a. If you do not want to install all IBM WebSphere MQ components, open **MQSERIES**
      1) Mark the components you want to install. The installer resolves dependencies automatically.
      2) Review the information displayed by the installer.

5. Optional: To install IBM WebSphere MQ to a non-default location, select **Actions > Change Product Location**. The default installation location is `/opt/mqm`.

   For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location.

   The installation path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.

   Note: Ensure that you do not select **Actions > Change Target** by accident, they are not the same.
6. If this installation is not the first installation on the system, select **Options > Allow creation of multiple versions**
7. Select **Actions > Install**. The log file tells you if there are any problems that need fixing.
8. Fix any problems, and click **OK** to install. You are informed when the installation has finished.
9. If this installation is not the first installation on the system, you must enter the following command to configure IBM WebSphere MQ:

   ```
   swconfig -x allow_multiple_versions=true MQSERIES,l=MQ_INSTALLATION_PATH
   ```

   where *MQ_INSTALLATION_PATH* is the path where you have just installed IBM WebSphere MQ. If you do not enter this command, the **swlist** command reports the installation as installed instead of configured. You must not use IBM WebSphere MQ unless the installation is configured.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  where *MQ_INSTALLATION_PATH* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Non-interactive installation of IBM WebSphere MQ server on HP-UX"
You can perform a non-interactive installation of IBM WebSphere MQ using the **swinstall** command. A non-interactive installation is also known as a silent, or unattended installation.

"Uninstalling IBM WebSphere MQ on HP-UX" on page 306
On HP-UX, you can uninstall theIBM WebSphere MQ server or client using the **swremove** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Non-interactive installation of IBM WebSphere MQ server on HP-UX:

You can perform a non-interactive installation of IBM WebSphere MQ using the **swinstall** command. A non-interactive installation is also known as a silent, or unattended installation.

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

About this task

This topic describes the non-interactive installation of a server, using the **swinstall** program to select which components you want to install. The components and are listed in "Choosing what to install" on page 159.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Set your current directory to the location of the installation file. The location might be the mount point of the CD, a network location, or a local file system directory.
3. Accept the IBM WebSphere MQ license agreement without an interactive prompt by entering the following command:

   ```
   ./mqlicense.sh -accept
   ```
4. Install WebSphere MQ using the **swinstall** command:
   a. If this installation is not the first installation on the system, you must add `-x allow_multiple_versions=true` to the **swinstall** command.
   b. Add the names of the components to install as parameters of the **swinstall** command. The installer automatically resolves any dependencies.
   c. Optional: Identify the installation location by adding `,l=`*`MQ_INSTALLATION_PATH`* as a parameter of the **swinstall** command. For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location. The installation path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.

For example, to install all IBM WebSphere MQ components, in a non-default location, as the first installation, enter the following command:

```
swinstall -s /installation_file.v11 MQSERIES,l=/opt/customLocation
```

To perform a partial installation, providing a list of components, in the default location, as the second installation, enter the following command:

```
swinstall -x allow_multiple_versions=true -s /installation_file.v11
MQSERIES.MQM-RUNTIME MQSERIES.MQM-BASE MQSERIES.MQM-SERVER
```

/installation_file.v11 is the absolute path to the installation file. The path must begin with a / and end with the name of the installation file. The installation file has the extension .v11.

5. If this installation is not the first installation on the system, you must enter the following command to configure the installation:

   ```
   swconfig -x allow_multiple_versions=true MQSERIES,l=MQ_INSTALLATION_PATH
   ```

   where *MQ_INSTALLATION_PATH* is the path where you have just installed IBM WebSphere MQ. If you do not enter this command, the **swlist** command reports the installation as installed instead of configured. You must not use IBM WebSphere MQ unless the installation is configured.

Example

The example shows the command to run a silent, full installation in the default location. Notice that all the language features are installed. Run a partial installation to install your chosen languages.

```
cd /downloads/WMQInstallFiles
swinstall -v -x source_directory=$PWD/hpUxxxxx.v11 MQSERIES
```

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  where *MQ_INSTALLATION_PATH* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Installing IBM WebSphere MQ server on HP-UX" on page 214

"Uninstalling IBM WebSphere MQ on HP-UX" on page 306
On HP-UX, you can uninstall theIBM WebSphere MQ server or client using the **swremove** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Installing IBM WebSphere MQ server on Linux

You can install a IBM WebSphere MQ server on either a 32-bit or 64-bit Linux system.

Before you begin

- Before you start the installation procedure, ensure that you have completed the necessary steps outlined in "Preparing the system" on page 196.
- If this installation is not the first installation on the system, you must ensure that the **crtmqpkg** command can write to a temporary location. By default, the **crtmqpkg** command will write to the `/var/tmp` directory. To use a different location, you can set the *TMPDIR* environment variable before you run the **crtmqpkg** command.
- To run the **crtmqpkg** command used in this task, you must have the **pax** command or **rpmbuild** installed.

  Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

About this task

Install the server by using the RPM Package Manager installer to select the components you want to install. The components and package names are listed in "Choosing what to install" on page 159.

Procedure

1. Log in as `root`, or switch to the superuser by using the **su** command.
2. Set your current directory to the location of the installation file. The location might be the mount point of the server DVD, a network location, or a local file system directory.
3. Run the `mqlicense.sh` script. If you want to view a text-only version of the license, which can be read by a screen reader, type the following message:

   ```
   ./mqlicense.sh -text_only
   ```

   The license number is displayed.

   You must accept the license agreement before you can proceed with the installation.
4. If this installation is not the first installation of IBM WebSphere MQ on the system, you must run the **crtmqpkg** command to create a unique set of packages to install on the system. To run the **crtmqpkg** command to run on Linux, you must install the **pax** command and **rpmbuild**, which is located in the rpm-build package.

   Note: The **crtmqpkg** command is required only if this is not the first installation of IBM WebSphere MQ on the system. If you have earlier versions of IBM WebSphere MQ installed on your system then installing IBM WebSphere MQ version 7.5 will work correctly if you install it in a different location.

   To run the **crtmqpkg** command on a Linux system:

   a. Enter the following command:

      ```
      ./crtmqpkg suffix
      ```

      where *suffix* is a name of your choosing that uniquely identifies the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.

      Note: This command creates a full copy of the installation packages in a temporary directory. By default, the temporary directory is located at `/var/tmp`. You must ensure that the system has enough free space before you run this command. To use a different location, you can set the *TMPDIR* environment variable before you run the **crtmqpkg** command. For example:

```
$ TMPDIR=/test ./crtmqpkg
```

b. Set your current directory to the location specified when the **crtmqpkg** command operation completes successfully. This directory is a subdirectory of the /var/tmp/mq_rpms directory, in which the unique set of packages is created. The packages have the *suffix* value contained within the file name. In the following example, the suffix of "1"./crtmqpkg 1 means that there is :a subdirectory named /var/tmp/mq_rpms/1/i386.

The packages are renamed according to the subdirectory, for example:

```
From: MQSeriesRuntime-7.5.0-0.i386.rpm
To: MQSeriesRuntime_1-7.5.0-0.i386.rpm
```

5. Install IBM WebSphere MQ. At a minimum, you must install the MQSeriesRuntime and the MQSeriesServer components.
   - To install to the default location, /opt/mqm, use the **rpm -ivh** command to install each component that you require.

     For example, to install the runtime and server components to the default location, use the following command:

     ```
     rpm -ivh MQSeriesRuntime-*.rpm MQSeriesServer-*.rpm
     ```

     To install all components to the default location use the following command:

     ```
     rpm -ivh MQSeries*.rpm
     ```

   - To install to a non-default location, use the **rpm --prefix** option. For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location.

     The installation path specified must be either an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.

     For example, enter the following installation path to install the runtime and server components to the /opt/customLocation directory on a 32-bit Linux system:

     ```
     rpm --prefix /opt/customLocation -ivh MQSeriesRuntime_*.rpm
     MQSeriesServer_*.rpm
     ```

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  where *MQ_INSTALLATION_PATH* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Uninstalling IBM WebSphere MQ on Linux" on page 307
On Linux, you can uninstall theIBM WebSphere MQ server or client using the **rpm** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Installing IBM WebSphere MQ server on Linux Ubuntu

You can install a IBM WebSphere MQ server on a Linux Ubuntu system.

Before you begin

- Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.
- Ensure that RPM is installed on your system, as the installation procedure uses the same RPM packages used by the other RPM based distributions. Technologies that convert these RPM packages into other forms, such as alien to convert RPMs to Debian packages, are not compatible with the IBM WebSphere MQ RPM packages and must not be used.

  IBM WebSphere MQ for Linux is installed by using RPM, which is not installed by default on Ubuntu. To determine if RPM is installed on you system, use the **dpkg** command. For example:

  ```
  dpkg -l rpm
  ```

  If the response from this command is of the form:

  ```
  ii  rpm      4.9.1.1-1ubuntu0.1     package manager for RPM
  ```

  Then RPM is already installed on your system and no further action is required.

  If the **dpkg** command returns output of the form:

  ```
  $ dpkg -l rpm
  No packages found matching rpm
  ```

  Then you must install the RPM package before you install IBM WebSphere MQ. For example:

  ```
  sudo apt-get install rpm
  ```

  If this command does not complete successfully, consult your Ubuntu administrator for instructions specific to your system to install the RPM package.

About this task

Install the server by using the RPM Package Manager installer to select the components you want to install. The components and package names are listed in "Choosing what to install" on page 159.

Procedure

1. Open a shell terminal and set your current directory to the location of the installation packages. The location might be the mount point of the server DVD, a network location, or a local file system

directory. You must have root authority to run the following commands. You can do so by adding **sudo** before the following commands, or by changing to the root user in the shell with the **su** command.

2. Run the mqlicense.sh script. If you want to view a text-only version of the license, which can be read by a screen reader, type the following message:

   ```
   ./mqlicense.sh -text_only
   ```

   The license is displayed.

   You must accept the license agreement before you can proceed with the installation.

3. If this installation is not the first installation of IBM WebSphere MQ version 7.5 on the system, you must run the **crtmqpkg** command to create a unique set of packages to install on the system. To run the **crtmqpkg** command to run on Linux, you must install the **pax** command and **rpmbuild**, which is located in the rpm-build package.

   Note: If you want to install IBM WebSphere MQ in a non-default installation location, apply the code fix provided in the technote, Installing WebSphere MQ on the Linux Ubuntu Distribution. Without this update, the **crtmqpkg** command will abort without creating the rebuilt RPM packages.

   a. Enter the following command:

      ```
      ./crtmqpkg suffix
      ```

      where *suffix* is a name of your choosing, that uniquely identifies the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.

      Note: This command creates a full copy of the installation packages in a temporary directory. By default, the temporary directory is located at /var/tmp. You must ensure that the system has enough free space before you run this command. To use a different location, you can set the *TMPDIR* environment variable before you run the **crtmqpkg** command. For example:

      ```
      TMPDIR=/test ./crtmqpkg
      ```

   b. Set your current directory to the location specified when the **crtmqpkg** command completes. This directory is a subdirectory of the /var/tmp/mq_rpms directory, in which the unique set of packages is created. The packages have the *suffix* value contained within the file name. In the following example, the suffix of "1"./crtmqpkg 1 means that there is :a subdirectory named /var/tmp/mq_rpms/1/i386.

      The packages are renamed according to the subdirectory, for example:

      ```
      From: MQSeriesRuntime-7.5.0-0.i386.rpm
      To: MQSeriesRuntime_1-7.5.0-0.i386.rpm
      ```

4. Install IBM WebSphere MQ. At a minimum, you must install the MQSeriesRuntime and the MQSeriesServer components.

   Two additional flags are required over the RPM based distribution installations:

   - **--nodeps**: You must use this option as the RPM database on Ubuntu is empty or sparsely populated, and the operating system package dependencies requested by the IBM WebSphere MQ packages are not listed within the RPM package database.

     Important: This option disables the safety checks contained within the IBM WebSphere MQ RPM packages to ensure that all WebSphere MQ packages contain their necessary prerequisite packages. If you are not installing all the components, you must ensure that appropriate sets are installed, as listed in Table 37 on page 222.
   - **--force-debian**: You must use this option to prevent warning messages from Ubuntu's version of RPM, which indicate that the RPM packages are not intended to be directly installed using RPM on Ubuntu.

As dependency checking is disabled during the Ubuntu IBM WebSphere MQ installation, install the MQSeriesRuntime package before any other package as all other packages depend on the MQSeriesRuntime package.

For example:

```
rpm -ivh --nodeps --force-debian MQSeriesRuntime-*.rpm
```

You can now choose other components to install, such as MQSeriesServer, MQSeriesJava, and MQSeriesJRE. These components provide your system with queue manager functionality, the IBM WebSphere MQ classes for Java and JMS APIs for using with Java applications, and a JRE to run the applications with. Use the following command:

```
rpm -ivh --nodeps --force-debian MQSeriesServer-*.rpm MQSeriesJRE-*.rpm MQSeriesJava-*.rpm
```

describes the package component dependencies. To install and use the package listed in the package name column you must also install the components listed in the package dependencies column.

Table 37. Package component dependencies

Package Name	Component Function	Package Dependencies
MQSeriesRuntime	Common function for all other components	None
MQSeriesServer	Queue Manager	MQSeriesRuntime
MQSeriesClient	C WebSphere MQ client libraries	MQSeriesRuntime
MQSeriesJava	Java and JMS WebSphere MQ APIs	MQSeriesRuntime
MQSeriesJRE	Java Runtime Environment	MQSeriesRuntime
MQSeriesExplorer	WebSphere MQ Explorer **Note:** There is no IBM support for this component on Ubuntu	MQSeriesRuntime MQSeriesJRE
MQSeriesGSKit	IBM Global Security Kit **Note:** There is no IBM support for this component on Ubuntu	MQSeriesRuntime MQSeriesJRE
MQSeriesSDK	Header files and libraries for non-Java APIs	MQSeriesRuntime
MQSeriesMan	UNIX man pages for WebSphere MQ	MQSeriesRuntime
MQSeriesSamples	WebSphere MQ application samples	MQSeriesRuntime

Table 37. Package component dependencies (continued)

Package Name	Component Function	Package Dependencies
MQSeriesMsg_cs MQSeriesMsg_de MQSeriesMsg_es MQSeriesMsg_fr MQSeriesMsg_hu MQSeriesMsg_it MQSeriesMsg_ja MQSeriesMsg_ko MQSeriesMsg_pl MQSeriesMsg_pt MQSeriesMsg_ru MQSeriesMsg_Zh_CN MQSeriesMsg_Zh_TW	Language specific message catalog files	MQSeriesRuntime
MQSeriesXRService	(64-bit Linux only) Telemetry queue manager component	MQSeriesRuntime MQSeriesJRE MQSeriesServer MQSeriesRuntime
MQSeriesXRClient	(64-bit Linux only) Telemetry client APIs	MQSeriesRuntime
MQSeriesFTBase	MQ Managed File Transfer component	MQSeriesRuntime MQSeriesJava MQSeriesJRE
MQSeriesFTLogger	MQ Managed File Transfer component	MQSeriesRuntime MQSeriesServer MQSeriesFTBase MQSeriesJava MQSeriesJRE
MQSeriesFTTools MQSeriesFTAgent	MQ Managed File Transfer components	MQSeriesRuntime MQSeriesFTBase MQSeriesJava MQSeriesJRE

Table 37. Package component dependencies (continued)

Package Name	Component Function	Package Dependencies
MQSeriesFTService	MQ Managed File Transfer component	MQSeriesRuntime MQSeriesServer MQSeriesFTAgent MQSeriesFTBase MQSeriesJava MQSeriesJRE
MQSeriesAMS	Advanced Message Security component **Note:** There is no IBM support for this component on Ubuntu	MQSeriesRuntime MQSeriesServer

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  where *MQ_INSTALLATION_PATH* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Uninstalling IBM WebSphere MQ on Linux" on page 307
On Linux, you can uninstall theIBM WebSphere MQ server or client using the **rpm** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Installing IBM WebSphere MQ server on Solaris

You can install an IBM WebSphere MQ server on Solaris either interactively or silently.

Before you begin

- Before you start the installation procedure, make sure that you complete the necessary steps that are outlined in "Preparing the system" on page 196.
- If you are using Solaris zones, you have a choice between installing IBM WebSphere MQ into the global zone, or installing IBM WebSphere MQ into a non-global zone. For details on how to install IBM WebSphere MQ into Solaris zones, see the following technote: WebSphere MQ V6.0 support position regarding Solaris zones. The technote is applicable to IBM WebSphere MQ Version 7.1 or later with the following changes:
  - You do not need the `-G` option on the **`pkgadd`** command as GSKit is now installed as part of the WebSphere MQ installation.
  - If you install IBM WebSphere MQ into the global zone for use in sparse zones, you must copy the `/var/mqm` file system into the sparse zone. You must also copy the `/etc/opt/mqm/mqinst.ini` installation entry into the sparse zone.
  - Limitations for shared `/usr` file systems: the **`dspmqinst`** and **`dspmqver`** commands might report the primary installation incorrectly when compared with the symbolic links in `/usr/bin`. To synchronize the reporting of the primary installation in a Solaris zone and the global zone, run **`setmqinst`** with the **`-i`** or **`-x`** parameter, on the individual zones.
  - You cannot change the primary installation within a non-global zone. You must change the primary installation through the global zone, which has the appropriate write access to `/usr/bin` .

About this task

This task describes the installation of the IBM WebSphere MQ for Solaris server, using the **pkgadd** program. You can choose which components you want to install. The components are listed in "Choosing what to install" on page 159.

Note: If you are installing on the Solaris 11 operating system, ensure that the IPS package (package/svr4) that supports **pkgadd** and equivalent utilities is installed.

Procedure

1. Log in as root, or switch to the superuser using the **`su`** command.
2. Set your current directory to the location of the installation file. The location might be the mount point of the server DVD, a network location, or a local file system directory.
3. Run the `mqlicense.sh` script to accept the license:

   ```
   ./mqlicense.sh
   ```

   If you want to view a text-only version of the license, which can be read by a screen reader, type:

   ```
   ./mqlicense.sh -text_only
   ```

   The license is displayed. Follow the instructions to accept the license. If you accept the license, the installation continues. If you do not accept the license, you cannot continue the installation process.
4. If this installation is not the first installation on the system, you must run **`crtmqpkg`** to create a unique set of packages to install on the system:
   a. Enter the following command:

      ```
      ./crtmqpkg suffix
      ```

      where *suffix* is a name of your choosing that uniquely identifies the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.

      The **`ctrmqpkg`** script can use two environment variables which are useful when you are installing from a non-disk media location:

      - *CDROOT*, the root of the installation media or downloaded installation files.

- *TMPDIR*, the output location of the modified installation files.

  No environment variables are required if you are running the image as `./crtmqpkg`.

b. Set your current directory to the location specified when the **`crtmqpkg`** command completes. This directory is a subdirectory of `/var/spool` , in which the unique set of packages is created. The packages have the *suffix* value contained within the file name.

5. Start the installation process:
   - If the installation is the first installation on the system, enter the following command to start the installation process:
     ```
     pkgadd -d.
     ```
   - If the installation is not the first installation on the system, enter the following command to start the installation process:
     ```
     pkgadd mqm-suffix
     ```

     where *suffix* is the suffix chosen in the previous step.
6. You are prompted to choose a location for installation.
   - To install to the default location, `/opt/mqm` , enter `y`.
   - To install to a non-default directory, enter `n` . Then, enter the required installation path, and confirm your choice.
7. When the list of components is displayed, enter the numbers of the components that you require, separated by spaces or commas. When you are installing (adding) an IBM WebSphere MQ component to an existing installation, choose option `yes` when you are asked whether to overwrite.

   Note: During the IBM WebSphere MQ base version installation, you can choose to install all components or a subset of the components. When you install a fix pack, only the currently installed components are upgraded. If, at a later stage, you want to add further IBM WebSphere MQ components that are not already installed, these components can be installed (added) to the IBM WebSphere MQ base version only. If your current version of IBM WebSphere MQ is not the base version, you must first uninstall all the fix packs before you add the required components to the existing installation, and then install the required fix packs. Also, when you are adding IBM WebSphere MQ components to an existing installation, you must choose option `yes` when you are asked whether to overwrite by the installation process.
8. If the path chosen in step 6 does not exist, you are asked if you want to create it. You must enter `y` to proceed.
9. Answer any questions appropriately for your system. If you are prompted to choose whether to install certain IBM WebSphere MQ files as `setuid/setgid` files, you must enter `y`.
10. A message is issued when the installation is complete. Enter `q` to exit the **pkgadd** program.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:
  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```
  where *`MQ_INSTALLATION_PATH`* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **`setmqenv`** or **`crtmqenv`** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Non-interactive installation of IBM WebSphere MQ server on Solaris"

"Uninstalling IBM WebSphere MQ on Solaris" on page 308
On Solaris, you can uninstall theIBM WebSphere MQ server or client using the **pkgrm** command.

Related information:

setmqinst

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Non-interactive installation of IBM WebSphere MQ server on Solaris:

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

About this task

You can perform a silent installation of IBM WebSphere MQ. A sample script file called `silent.sh` is supplied in the `silent` directory on the DVD. You can use this script to perform a non-interactive installation that requires no input and shows nothing on the screen. It must be run as root.

The installation script `silent.sh` uses an `admin` file and a `response` file, both of which are supplied in the `silent` directory. You can use these files as supplied to perform a silent installation of all the components, including all the national language features, to the default location.

Note: If you are installing on the Solaris 11 operating system, ensure that the IPS package (package/svr4) that supports **pkgadd** and equivalent utilities is installed.

Procedure

1. Copy the `silent.sh` script into a writeable directory.
2. If this installation is not the first installation on the system, you must run **crtmqpkg** to create a unique set of packages to install on the system:
   a. Enter the following command:
   ```
   ./crtmqpkg suffix
   ```
   where *suffix* is a name of your choosing, that will uniquely identify the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.
   b. Set your current directory to the location specified when the **crtmqpkg** command completes. This directory is a sub-directory of `/var/spool`, in which the unique set of packages is created. The packages have the *suffix* value contained within the filename.

Once a new package has been generated for the second installation the `silent.sh` script needs to have its MQ_PACKAGE_NAME variable modified so that its value is not `mqm` but the new package name.

Also the MQ_PACKAGE_LOCATION variable needs to be modified so that its value is not $MQ_MEDIA_LOCATION but the location of the new package (which by default is `/var/spool/pkg`).

3. Optional: If you want to change where the IBM WebSphere MQ server DVD is mounted, you must update the values in the `silent.sh` script. By default, the script assumes that the server DVD has been mounted at `/CD7FVML`.
4. Optional: If you want to change where the output and logs are written to, update the values in the `silent.sh` script. By default, output and logs are written to the file `/var/tmp/mq.install`.
5. Optional: If you want to install to a non-default location, you must update the *`MQ_INSTALLATION_PATH`* variable in the `silent.sh` script.

   Note:

   - The installation path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.
   - The directory specified must exist before running the **`pkgadd`** command.
6. Optional: If you want to change the components that are installed, you must edit the `response` file. A list of all the installable IBM WebSphere MQ components can be found at: "Choosing what to install" on page 159.

   Solaris does not check, during a silent installation, that prerequisite components are installed. You can use the following procedure to create a response file interactively, before using it to install the product. **`pkgask`** prompts you for the names of the components to install.

   a. Run the **`mqlicense.sh`** command to accept the license agreement for the product.
   b. **pkgask** *-d path_to_install_image -r response_file* **mqm**

   The inputs to **`pkgask`** are the same as those inputs documented for **`pkgadd`**, but instead of the product being installed a response file is created.
7. Optional: If you have edited the `response` file, you must then edit the `silent.sh` to use your custom response file.
8. To start the installation, run `silent.sh`.
9. Check the log file for any errors.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  where *`MQ_INSTALLATION_PATH`* represents the directory where IBM WebSphere MQ is installed.

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **`setmqenv`** or **`crtmqenv`** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- If you want to confirm that the installation was successful, you can verify your installation. See "Verifying a server installation" on page 280, for more information.

Related concepts:

"Multiple installations" on page 150
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

"Choosing a primary installation" on page 152
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Related tasks:

"Installing IBM WebSphere MQ server on Solaris" on page 224
You can install an IBM WebSphere MQ server on Solaris either interactively or silently.

"Uninstalling IBM WebSphere MQ on Solaris" on page 308
On Solaris, you can uninstall theIBM WebSphere MQ server or client using the **`pkgrm`** command.

Related information:

setmqinst

Changing the primary installation
You can use the **`setmqinst`** command to set or unset an installation as the primary installation.

Installing IBM WebSphere MQ server on Windows

This topic describes how to install IBM WebSphere MQ server on Windows systems by using the Launchpad. This procedure can be used for installing a first or a subsequent installation.

Installing using the Launchpad:
About this task

These instructions cover how to display the installation Launchpad window. You can use the launchpad to make a compact, typical, or custom installation of IBM WebSphere MQ. You can reuse the launchpad multiple times to install further installations. It automatically selects the next available installation name, instance, and location to use. To view all the installation types and the features that are installed with each option, see "Planning your installation on Windows systems" on page 181.

Note that if you have previously uninstalled IBM WebSphere MQ from your system (see "Uninstalling IBM WebSphere MQ on Windows systems" on page 309), some configuration information might remain, and some default values might be changed.

Procedure

1. Access the IBM WebSphere MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `setup.exe` in the base directory of the IBM WebSphere MQ installation image.
   - From a DVD, this location might be `E:\setup.exe`
   - From a network location, this location might be `m:\instmqs\setup.exe`
   - From a local file system directory, this location might be `C:\instmqs\setup.exe`
3. Double-click the **Setup** icon to start the installation process. It is possible to run either by:
   - Running `setup.exe` from the command prompt. Or
   - Double-clicking `setup.exe` from Windows Explorer.

   If you are installing on a Windows system with UAC enabled, accept the Windows prompt to allow the launchpad to run as elevated. During installation, you might also see Open File - Security Warning dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the installation to continue.

   The IBM WebSphere MQ Installation Launchpad window is displayed.
4. Continue to follow the Launchpad instructions as shown on screen.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Known problem: The installation of IBM WebSphere MQ on Windows might fail while installing Microsoft Visual C++ runtime due to a conflict between the runtime merge modules that are installed by IBM WebSphere MQ and the run time that is already installed on the machine by another product.

If this issue occurs, a message like the one shown in the following example is recorded in the IBM WebSphere MQ installation logs:

```
MSI (s) (34:48) [12:07:26:083]: Product: IBM WebSphere MQ 64bitSupport -- Error 1935.
An error occurred during the installation of assembly component {844EFBA7-1C24-93B2-A01F-C8B3B9A1E18E}. HRESULT: 0x80073715
```

This problem is due to a known Microsoft issue that is documented in the following Microsoft Support article: http://support.microsoft.com/kb/2643995.

The solution is to download and install latest Microsoft Visual C++ 2005 from http://www.microsoft.com/en-us/download/details.aspx?id=26347 and then try the IBM WebSphere MQ installation again.

Related concepts:

"Modifying your installation" on page 244
Find out how to modify a IBM WebSphere MQ server installation interactively using the launchpad or non-interactively using msiexec.

"Post installation tasks" on page 245
Find out what tasks can be undertaken once IBM WebSphere MQ has been successfully installed. Begin by following the related pages below:

Related tasks:

"Advanced installation using msiexec" on page 231

Installing IBM WebSphere MQ using SCCM:

You can install IBM WebSphere MQ using the System Center Configuration Manager (SCCM).

Procedure

1. Create a folder on the SCCM server and copy the IBM WebSphere MQ DVD image into that folder.
2. Make the folder the X drive of the SCCM client systems.
3. Using the Configuration Manager Console for SCCM, create a package:
   a. Click **Computer Management > Software Distribution > Packages**.
   b. Right-click on **Packages** and click **New > Package from Definition**.
   c. In the wizard, select the `WebSphere MQ.SMS` file from the folder that you copied the IBM WebSphere MQ DVD image into. If you chose to use a drive letter other than X for the client systems, you must modify the commands in the `WebSphere MQ.SMS` file to use the appropriate drive letter.

4. Optional: To install IBM WebSphere MQ to a non-default location, you must add the following two parameters to the command line under Program Properties:

   ```
   PGMFOLDER="MQ_INSTALLATION_PATH"
   DATFOLDER="MQ_INSTALLATION_PATH\data"
   ```

   where *MQ_INSTALLATION_PATH* refers to the path where you want to install IBM WebSphere MQ.
5. Under the package you created, Right click **Distribution Points** and then click **New distribution Points**.
6. In the wizard, select the SCCM server as the distribution point.
7. Using the Configuration Manager Console for SCCM, distribute the software:
   a. Right-click the package you created and click **Distribute > Software**.
   b. In the wizard, select **All Systems for Collections**.
8. Once the Advertisement Status for the package shows as Received, manually initiate the IBM WebSphere MQ unattended installation:
   a. Log on to the SCCM client system
   b. Click **Control Panel > Run Advertised Program** and select the package to run

Advanced installation using msiexec:
Before you begin

If you are running IBM WebSphere MQ on Windows systems with User Account Control (UAC) enabled, you must invoke the installation with elevated privileges. If you are using the Command prompt or IBM WebSphere MQ Explorer elevate privileges by using a right-click to start the program and selecting Run as administrator. If you try to run **msiexec** without using elevated privileges, the installation fails with an error of AMQ4353 in the installation log.

About this task

WebSphere MQ on Windows uses the MSI technology to install software. MSI provides both an interactive installation and a non interactive installation. An interactive installation displays panels and ask questions.

The **msiexec** command uses parameters to give MSI some or all of the information that can also be specified through panels during an interactive installation. This means that a user can create a reusable automated or semi-automated installation configuration. Parameters can be given through the command line, a transform file, a response file, or a combination of the three.

Procedure

To install using msiexec, at the command line, enter the **msiexec** command in the following format:

```
msiexec parameters [USEINI="response-file"] [TRANSFORMS="transform_file"]
```

Where:

parameters
: are either command-line parameters preceded by a / character, or property=value pairs (if using both forms of parameter always put the command-line parameters first). For further information, see “Specifying command line parameters with msiexec” on page 232, which contains a link to the web site that lists all the command line parameters that are available.

  For an unattended installation, you must include the /q or /qn parameter in the command line. Without this parameter, the installation is interactive.

  Note: You must include the **/i** parameter and the file location of the IBM WebSphere MQ installer package.

response-file
: is the full path and file name of the file that contains the [Response] stanza and the required property=value pairs, for example `C:\MyResponseFile.ini`. An example response file, `Response.ini`, is supplied with IBM WebSphere MQ. This file contains default installation parameters. For further information, see "Using a response file with msiexec" on page 233.

transform_file
: is the full path and file name of a transform file. For further information, see "Using transforms with msiexec" on page 237 and "Multiple installation using MSI Instance ID" on page 237.

Note: For a silent installation to succeed, the AGREETOLICENSE=?YES? property must be defined either on the command line or in the response file.

Results

After the command has been entered, the command prompt immediately reappears. IBM WebSphere MQ is installing as a background process. If you have entered parameters to produce a log, check this file to see how the installation is progressing. If the installation completes successfully, you see the message `Installation operation completed successfully` in the log file.

Specifying command line parameters with msiexec:

About this task

The **msiexec** command can accept two types of parameters on the command line, as follows:

- Standard command line parameters, preceded by a / character.

  For a table of the **msiexec** command line parameters, see the MSDN Command-Line Options web page.
- Property=value pair parameters on the command line. All the parameters available for use in a response file can be used on the command line, for a list of these, see Table 39 on page 234. In addition there are some extra property=value pair parameters that are only for use on the command line, for details see Table 38 on page 233.

  When using the property=value pair parameters note that:

  - Property strings must be in uppercase.
  - Value strings are not case-sensitive, except for feature names. You can enclose value strings in double quotation marks. If a value string includes a blank, enclose the blank value string in double quotation marks.
  - For a property that can take more than one value, use the format:

    ```
    ADDLOCAL="Server,Client"
    ```

  When using property=value pair and command line parameters with the **msiexec** command, enter command line parameters first.

  If a parameter is specified both on the command line and in a response file, the setting on the command line takes precedence.

Example

Here is an example of a typical **msiexec** command. All parameters, separated by one or more spaces, must be typed on the same line as the **msiexec** call.

```
msiexec
 /i "path\MSI\IBM WebSphere MQ.msi"
 /l*v c:\install.log
 /q
 TRANSFORMS="1033.mst"
 AGREETOLICENSE="yes"
 ADDLOCAL="Server"
```

Here is an example of a typical **msiexec** command when you are installing a second copy of IBM WebSphere MQ Version 7.5. All parameters, separated by one or more spaces, must be typed on the same line as the **msiexec** call.

```
msiexec
 /i "path\MSI\IBM WebSphere MQ.msi"
 /l*v c:\install.log
 /q
 TRANSFORMS=":InstanceId2.mst;1033.mst"
 AGREETOLICENSE="yes"
 ADDLOCAL="Server"
 MSINEWINSTANCE=1
```

Where `/l*v c:\install.log` writes installation log to file `c:\install.log`.

The following table shows the parameters which can only be provided on the command line and not in a response file.

Table 38. msiexec property=value parameters

Property	Values	Meaning
USEINI	*path\file_name*	Use the specified response file. See "Using a response file with msiexec"
SAVEINI	*path\file_name*	Generate a response file during installation. The file contains those parameters selected for this installation that a user might make during an interactive installation.
ONLYINI	1|yes|""	1, yes or any value other than null. End the installation before updating the target system, but after generating a response file, if this is specified. "". Continue the installation and update the target system (the default).
TRANSFORMS	:InstanceId*x*.mst|*path\file_name*| :InstanceId*x*.mst;*path\file_name*	The :InstanceId*x*.mst value is only required for a subsequent installation of IBM WebSphere MQ Version 7.5. The *path\file_name* specifies what transform (.mst) files must be applied to the product. For example, `"1033.mst"` specifies the supplied U.S. English transform file.
MSINEWINSTANCE	1	This property is only required for subsequent installations of IBM WebSphere MQ Version 7.5

Using a response file with msiexec:

About this task

You can use the **msiexec** command with a parameter which specifies additional properties are defined in a response file. You can combine the `msiexec` command-line parameters described in "Specifying command line parameters with msiexec" on page 232.

A response file is an ASCII text file, with a format like a Windows `.ini` file, that contains the stanza [Response]. The [Response] stanza contains some or all the parameters that would normally be specified as part of an interactive installation. The parameters are given in a property=value pair format. Any other stanzas in the response file are ignored by **msiexec**. An example response file, `Response.ini`, is supplied with IBM WebSphere MQ. It contains the default installation parameters.

Procedure

A typical example of an `msiexec` command is: `msiexec /i "path\MSI\IBM WebSphere MQ.msi" /l*v c:\install.log    TRANSFORMS="1033.mst" USEINI="C:\MQ\Responsefile"`

If a parameter is specified both on the command line and in a response file, the setting on the command line takes precedence. All the parameters available for use in a response file can also be used on the command line, for a list of these see Table 39.

In the response file, all text is in English, and comments begin with a **;** character.

For information about creating a response file, see "Creating a response file" on page 238.

Example

An example of a typical response file:

```
[Response]
PGMFOLDER="c:\mqm"
DATFOLDER="c:\mqm\data"
LOGFOLDER="c:\mqm\log"
AGREETOLICENSE="yes"
LAUNCHWIZ=""
WIZPARMFILE="d:\MQParms.ini"
ADDLOCAL="Server,Client"
REMOVE="Toolkit"
```

Table 39. Response file parameters

Property	Values	Meaning
PGMFOLDER	*path*	Folder for the IBM WebSphere MQ program files. For example, `c:\mqm`.
DATFOLDER	*path*	Folder for the IBM WebSphere MQ data files. For example, `c:\mqm\data`. **Note:** Multiple installations of IBM WebSphere MQ all use the same **DATFOLDER**.
LOGFOLDER	*path*	Folder for the IBM WebSphere MQ queue manager log files. For example, `c:\mqm\log`. **Note:** Multiple installations of IBM WebSphere MQ all use the same **LOGFOLDER**.
USERCHOICE	0|no	If the command line or response file specifies parameters to install features, a dialog can be displayed to prompt the user to accept the preselected options, or review and possibly change them. 0 or no. Suppresses display of the dialog. Anything else. Dialog is displayed. Not used for a silent installation.
AGREETOLICENSE	yes	Accept the terms of the license. Set to yes before a silent installation. If the installation is not silent, this parameter is ignored.

Table 39. Response file parameters (continued)

Property	Values	Meaning
KEEPQMDATA	**keep** |delete	If the Server feature is to be uninstalled, whether to delete any existing queue managers. delete removes any existing queue managers. keep, or any other value, keeps them. **Note:** This property is only valid on a final server uninstall. If used on any other server unistalls this property is ignored.
LAUNCHWIZ	0|1|yes|no|**""**	0 or no. Do not launch the Prepare IBM WebSphere MQ wizard after IBM WebSphere MQ is installed. 1 or yes. Launch the Prepare IBM WebSphere MQ wizard if the Server feature is installed. "". Launch the Prepare IBM WebSphere MQ wizard to install the Server (the default). If this option is to launch the Prepare IBM WebSphere MQ wizard, you can specify the WIZPARMFILE, either in this file, or on the command line. The Prepare WebSphere MQ wizard must be run to make your WebSphere MQ installation operational. If you choose not to launch it here, you must run it before using WebSphere MQ.
WIZPARMFILE	*path\file_name*	When specified, the file that contains the parameters to pass to the Prepare IBM WebSphere MQ wizard when it is launched. These are in the [Services].
ADDLOCAL	*feature, feature, All* | ””	A comma-separated list of features to install locally. For a list of valid feature names, see “IBM WebSphere MQ features for Windows systems” on page 173. All installs all features "" installs the typical features. If you do not want a feature use REMOVE="*feature*" **Note:** If this is a new installation the typical features (Server, Explorer, Java Messaging and SOAP Transport, and Development Toolkit) are installed by default irrespective of the feature list provided in the ADDLOCAL property. If you do not want a feature use REMOVE="feature"
REMOVE	*feature, feature,* |All|**""**	A comma-separated list of features to remove. For a list of valid feature names, see “IBM WebSphere MQ features for Windows systems” on page 173. All uninstalls all features "" uninstalls no features (the default).

Table 39. Response file parameters (continued)

Property	Values	Meaning
STARTSERVICE	0 | no | ""	0 or no. Do not start the IBM WebSphere MQ Service at the end of installation. "" (the default). Start the IBM WebSphere MQ Service at the end of installation if it was running at the start, or if this is a new installation. Anything else. Start the Service at the end of the installation. Ignored if the server feature is not installed. If you do not start the WebSphere MQ Service, WebSphere MQ will not be operational and queue managers will not start. You must run the Prepare WebSphere MQ wizard for the service to be correctly configured. This parameter is only valid if LAUNCHWIZ is set to no.
STARTTASKBAR	0 | no | ""	0 or no. Do not start the IBM WebSphere MQ taskbar application at the end of installation. "" (the default). Start the IBM WebSphere MQ taskbar application at the end of installation if it was running at the start, or if this is a new installation. Anything else. Start the taskbar application at the end of the installation. Ignored if the server feature is not installed. This parameter is only valid if LAUNCHWIZ is set to no.
INSTALLATIONDESC	?Description of installation?	Sets the installation description from the command line. Subject to the documented installation description length limitations
INSTALLATIONNAME	[INSTALLATION0,]?Name?	Sets the installation name from the command line. Subject to the documented installation name character and length limitations. **Note:** Supply INSTALLATION0,Name only when upgrading from versions of IBM WebSphere MQ before Version 7.5.
MAKEPRIMARY	0 | 1 | ""	Makes the installation primary, if possible, or removes the primary flag. 1 = Make primary, 0 = Make non-primary, - use default algorithm **Note:** This option is ignored if a version of IBM WebSphere MQ before Version 7.5 is installed, or if another installation of IBM WebSphere MQ Version 7.5 is present and set as the primary.

Related tasks:

"Multiple installation using MSI Instance ID"
This topic describes how to choose the MSI instance ID you require for non-interactive multiple installations.

"Creating a response file" on page 238
A response file is used with **msiexec**. You can create it in three ways.

"Using the MQParms command" on page 239

Related reference:

"Using transforms with msiexec"

Multiple installation using MSI Instance ID:

This topic describes how to choose the MSI instance ID you require for non-interactive multiple installations.

About this task

In order to support non-interactive multiple installations, you need to find out whether the instance ID you want to use is already in use or not and choose the appropriate one. For each installation media (for example, each 7.5 client and 7.5 server), Instance ID 1 is the default ID which is used for single installations. If you want to install alongside Instance ID 1 you need to specify which instance you want to use. If you have already installed instance 1, 2, and 3 then you need to find out what the next available instance is, for instance, Instance ID 4. Similarly, if instance 2 has been removed, you need to find out that there is a gap that can be reused. You can find out which Instance ID is currently in use using the **dspmqinst** command.

Procedure

1. Type **dspmqinst** to find a free MSI Instance in the media being installed by reviewing the MSIMedia and MSIInstanceId values for the versions already installed. For example:

   ```
   InstName:      Installation1
   InstDesc:
   Identifier:    2
   InstPath:      C:\mq\install1
   Version:       7.5.0.0
   Primary:       Yes
   State:         Available
   MSIProdCode:   {0730749B-080D-4A2E-B63D-85CF09AE0EF0}
   MSIMedia:      7.5 Server
   MSIInstanceId: 1
   ```

2. If MSI Instance ID 1 is in use and you want to use MSI Instance ID 2, the following parameters must be added to the msiexec call:

   ```
   MSINEWINSTANCE=1 TRANSFORMS=:InstanceId2.mst
   ```

What to do next

For multiple installations, the **INSTALLATIONNAME** or **PGMFOLDER** must be supplied as an additional parameter on any non-interactive installation command. Supplying the **INSTALLATIONNAME** or **PGMFOLDER** ensures that you do not work with the wrong installation in case you omit or incorrectly specify the **TRANSFORMS** parameter.

Using transforms with msiexec:
MSI can use transforms to modify an installation. During IBM WebSphere MQ installation, transforms can be used to support different national languages. IBM WebSphere MQ is supplied with transform files in the `\MSI` folder of the Server image. These files are also embedded in the IBM WebSphere MQ Windows installer package, `IBM WebSphere MQ.msi`.

On the **msiexec** command line, you can specify the required language by using the TRANSFORMS property in a property=value pair. For example:

```
TRANSFORMS="1033.mst"
```

You can also specify the full path and file name of the transform file. Again, the quotation marks surrounding the value are optional. For example:

```
TRANSFORMS="D:\Msi\1033.mst"
```

Table 40 shows the locale identifier, language, and the transform file name to use in the **msiexec** command line.

You might need to merge transforms to install multiple installations of the same version, for example:

```
TRANSFORMS=":InstanceId2.mst;D:\Msi\1033.mst"
```

You can also specify the required language by using the MQLANGUAGE property with the **MQParms** command. For information about the msiexec property=value parameters, see "MQParms parameter file" on page 240.

Parameters

Table 40. Supplied transform files for various language support. This table shows the supplied transform files, the resulting language, and the numeric value to use in the **msiexec** command line.

Language	Transform File name	Value
U.S. English	1033.mst	1033
German	1031.mst	1031
French	1036.mst	1036
Spanish	1034.mst	1034
Italian	1040.mst	1040
Brazilian Portuguese	1046.mst	1046
Japanese	1041.mst	1041
Korean	1042.mst	1042
Simplified Chinese	2052.mst	2052
Traditional Chinese	1028.mst	1028
Czech	1029.mst	1029
Russian	1049.mst	1049
Hungarian	1038.mst	1038
Polish	1045.mst	1045

Creating a response file:

A response file is used with **msiexec**. You can create it in three ways.

About this task

A response file is used with the **msiexec** command, for further information see "Using a response file with msiexec" on page 233.

Procedure

There are three ways to create a response file for installation:

- Copy and edit the file `Response.ini` that is supplied on the IBM WebSphere MQ Windows Server CD, using an ASCII file editor.
- Create your own response file using an ASCII file editor.
- Use the **msiexec** command with the **SAVEINI** (and optionally, the **ONLYINI**) command line parameters to generate a response file that contains the same installation options. See Table 38 on page 233.

Example

A typical example of using **msiexec** with the **SAVEINI** parameter is here:

```
msiexec /i "path\IBM WebSphere MQ.msi" /q SAVEINI="response_file"
TRANSFORMS="1033.mst" AGREETOLICENSE="yes"
```

Using the MQParms command:
Before you begin

You can use the `MQParms` command to invoke installation or uninstallation. This command can use parameters on a command line, or those specified in a parameter file. The parameter file is an ASCII text file that contains the parameter values that you want to set for the installation. The `MQParms` command takes the specified parameters and generates the corresponding `msiexec` command line.

This means that you can save all the parameters that you want to use with the `msiexec` command in a single file.

If you are running IBM WebSphere MQ on Windows systems with User Account Control (UAC) enabled, you must invoke the installation with elevated privileges. If you are using the Command prompt or IBM WebSphere MQ Explorer elevate privileges by using a right-click to start the program and selecting **Run as administrator**. If you try to run the `MQParms` program without using elevated privileges, the installation fails with an error of AMQ4353 in the installation log.

For silent operations, this must include the **/q** or **/qn** parameter, either on the command line, or in the [MSI] stanza of the parameter file. You must also set the AGREETOLICENSE parameter to "yes".

You can specify many more parameters in the parameter file that you use with the `MQParms` command than you can in the response file that you use directly with the `msiexec` command. Also, as well as parameters that the IBM WebSphere MQ installation uses, you can specify parameters that can be used by the Prepare IBM WebSphere MQ wizard.

If you do not complete the Prepare WebSphere MQ Wizard directly after IBM WebSphere MQ installations or if for any reason your machine is rebooted between completing IBM WebSphere MQ installation and completing the Prepare WebSphere MQ Wizard, ensure that the wizard is run with Administrator privilege afterward, otherwise the installation is incomplete, and might fail. You might also see Open File - Security Warning dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the wizard to continue

An example of the file MQParms.ini is supplied with IBM WebSphere MQ. This file contains default installation parameters.

There are two ways to create a parameter file for installation:
- Copy and edit the file MQParms.ini that is supplied with the product, using an ASCII file editor.
- Create your own parameter file using an ASCII file editor.

About this task

To invoke installation using the `MQParms` command:

Procedure

1. From a command line, change to the root folder of the IBM WebSphere MQ Server DVD (that is, the location of the file MQParms.exe).
2. Enter the following command:

   ```
   MQParms parameter_file parameters]
   ```

   where:

   parameter_file
   : is the file that contains the required parameter values. If this file is not in the same folder as MQParms.exe, specify the full path and file name. If you do not specify a parameter file, the default is `MQParms.ini`. For a silent install, the `MQParms_silent.ini` parameter file can be used. For further details, see "MQParms parameter file."

   parameters
   : are one or more command-line parameters, for a list of these, see the MSDN Command-Line Options web page.

Example

A typical example of an MQParms command is:

```
MQParms "c:\MyParamsFile.ini" /l*v c:\install.log
```

A typical example of an MQParms command when you are installing a second copy of IBM WebSphere MQ Version 7.5 is:

```
MQParms "c:\MyParamsFile.ini" /l*v c:\install.log TRANSFORMS=":InstanceId2.mst;1033.mst" MSINEWINSTANCE=1
```

Alternatively, TRANSFORMS and MSINEWINSTANCE can be specified in the MSI stanza of the parameter file.

If you specify a parameter both on the command line and in the parameter file, the setting on the command line takes precedence.

If you specify a parameter file, you might want to run the encryption utility before you use the `MQParms` command (see "Encrypting a parameter file" on page 243).

If you do not specify `/i`, `/x`, `/a`, or `/j`, MQParms defaults to standard installation using the IBM WebSphere MQ Windows Installer package, IBM IBM WebSphere MQ.msi. That is, it generates the following part of the command line:

```
/i "current_folder\MSI\IBM WebSphere MQ.msi"
```

If you do not specify a WIZPARMFILE parameter, MQParms defaults to the current parameter file. That is, it generates the following part of the command:

```
WIZPARMFILE="current_folder\current_parameter_file"
```

MQParms parameter file:

A parameter file is an ASCII text file that contains sections (stanzas) with parameters that can be used by the `MQParms` command. Typically, this is an initialization file such as MQParms.ini.

The `MQParms` command takes parameters from the following stanzas in the file:

[MSI] Contains general properties related to how the `MQParms` command runs and to the installation of IBM WebSphere MQ.

The properties that you can set in this stanza are listed in "Advanced installation using msiexec" on page 231, and Table 41 on page 241.

[Services]

Contains properties related to IBM WebSphere MQ account configuration, in particular, the user account required for IBM WebSphere MQ Services. If you are installing IBM WebSphere MQ on a network where the domain controller is on a Windows 2003 server, you probably need details of a special domain account. For further information, see "Configuring WebSphere MQ accounts" on page 249 and "Configuring WebSphere MQ with the Prepare WebSphere MQ wizard" on page 246.

The properties that you can set in this stanza are listed in Table 43 on page 242.

`MQParms` ignores any other stanzas in the file.

The stanza parameters are in the form property=value, where property is always interpreted as uppercase, but value is case sensitive. If a value string includes a blank, it must be enclosed in double quotation marks. Most other values can be enclosed in double quotation marks. Some properties can take more than one value, for example:

```
ADDLOCAL="Server,Client"
```

To clear a property, set its value to an empty string, for example:

```
REINSTALL=""
```

The following tables show the properties that you can set. The default is shown in bold.

For the [MSI] stanza, you can enter standard MSI command line options and properties. For example:

```
- /q
- ADDLOCAL="server"
- REBOOT=Suppress
```

Refer to Table 41, Table 42 on page 242, and Table 43 on page 242 for the properties used to install IBM WebSphere MQ.

Table 41 shows additional properties in the stanza that affect how the `MQParms` command runs, but that do not affect the installation.

Table 41. Properties used by MQParms in the MSI stanza

Property	Values	Description
MQPLOG	*path*|*file_name*	MQParms generates a text log file with the specified name and location.
MQPLANGUAGE	**system**|user| *transform_value*|existing	The installation language. system. Install using the language of the default system locale (the default). user. Install using the language of the default locale of the user. *transform_value*. Install using the language specified by this value. See Table 42 on page 242. existing. If MQ already exists on the system, the same language will be used by default, otherwise system is used.
MQPSMS	**0**|no	0 or no. MQParms does not wait for the `msiexec` command to end (the default). Any other value. MQParms waits for the `msiexec` command to end.

Table 41. Properties used by MQParms in the MSI stanza (continued)

Property	Values	Description
MQPINUSE	**0** | 1	If MQPINUSE is set to 1, MQParams continues installing even if IBM WebSphere MQ files are in use. If this option is used a reboot will be required to complete the installation.

Table 42. Valid values for the MQPLANGUAGE property

Language	Valid values		
U.S. English	English	en_us	1033
German	German	de_de	1031
French	French	fr_fr	1036
Spanish	Spanish	es_es	1034
Italian	Italian	it_it	1040
Brazilian Portuguese	Brazilian Portuguese	pt_br	1046
Japanese	Japanese	ja_jp	1041
Korean	Korean	ko_kr	1042
Simplified Chinese	Simplified Chinese	zh_cn	2052
Traditional Chinese	Traditional Chinese	zh_tw	1028
Czech	Czech	cs_cz	1029
Russian	Russian	ru_ru	1049
Hungarian	Hungarian	hu_hu	1038
Polish	Polish	pl_pl	1045

For the [Services] stanza, you can enter parameters in property=value format. You might want to encrypt the values in this stanza. See "Encrypting a parameter file" on page 243.

Table 43. Properties in the Services stanza

Property	Values	Description
USERTYPE	**local** | domain | onlydomain	The type of user account to use: **local** Creates a local user account. **domain** Creates a local user account. If this does not have the required security authorities, it uses the domain user account specified by DOMAINNAME, USERNAME, and PASSWORD. **onlydomain** Does not create a local user account, but immediately uses the domain user account specified by DOMAINNAME, USERNAME and PASSWORD. If any of these three properties are missing, a USERTYPE of local is assumed. The properties DOMAINNAME, USERNAME, and PASSWORD are required if USERTYPE is set to onlydomain.

Table 43. Properties in the Services stanza (continued)

Property	Values	Description
DOMAINNAME	*domain_name*[1]	The domain for the domain user account. Required if USERTYPE is set to domain or onlydomain.
USERNAME	*user_name*[1]	The user name for the domain user account. Required if USERTYPE is set to domain or onlydomain..
PASSWORD	*password*[1]	The password for the domain user account. Required if USERTYPE is set to domain or onlydomain.
1. Do not enclose this value in double quotation marks.		

A typical example of a parameter file is:

```
[MSI]
MQPLANGUAGE=1033
MQPLOG=%temp%\MQParms.log
MQPSMS=no
ADDLOCAL=Server
/m miffile
REMOVE=""
/l*v c:\install.log

[Services]
USERTYPE=domain
DOMAINNAME=mqm*df349edfcab12
USERNAME=mqm*a087ed4b9e9c
PASSWORD=mqm*d7eba3463bd0a3
```

Encrypting a parameter file:

About this task

Use the `setmqipw` utility to encrypt the DOMAINNAME, USERNAME, and PASSWORD values in the [Services] stanza of a parameter file, if they are not already encrypted. (These values might be encrypted if you have run the utility before.) `setmqipw` will also encrypt the QMGRPASSWORD and CLIENTPASSWORD values in the [SSLMigration] stanza of a parameter file.

This encryption means that, if you need a special domain account to configure IBM WebSphere MQ (see “Configuring WebSphere MQ accounts” on page 249), or you need to keep key database passwords secret, details are kept secure. Otherwise, these values, including the domain account password, flow across the network as clear text. You do not have to use this utility, but it is useful if security in your network is an issue.

To run the script:

Procedure

1. From a command line, change to the folder that contains your parameter file.
2. Enter the following command:

   ```
   CD_drive:\setmqipw
   ```

   Note: You can run the command from a different folder, by entering the following command, where *parameter_file* is the full path and file name of the parameter file:

   ```
   CD_drive:\setmqipw parameter_file
   ```

Results

If you view the resulting parameter file, the encrypted values start with the string mqm*. Do not use this prefix for any other values; passwords or names that begin with this prefix are not supported.

The utility creates a log file, setmqipw.log, in the current directory. This file contains messages related to the encryption process. When encryption is successful, messages are similar to:

```
Encryption complete
Configuration file closed
Processing complete
```

What to do next

After you encrypt the parameter file, you can use it in the normal way with the MQParms command (see "Using the MQParms command" on page 239).

Modifying your installation:

Find out how to modify a IBM WebSphere MQ server installation interactively using the launchpad or non-interactively using msiexec.

Related concepts:

"Silently modifying a WebSphere MQ server installation using msiexec" on page 245

Related tasks:

"Modifying the installation using WebSphere MQ Installation Launchpad"

Modifying the installation using WebSphere MQ Installation Launchpad:

Before you begin

To modify an installation, some features of IBM WebSphere MQ must already be installed.

About this task

To remove or install IBM WebSphere MQ features follow the instructions. This procedure is the only way to interactively remove or install IBM WebSphere MQ features on Windows Vista and Windows Server 2008:

Procedure

1. Insert the IBM WebSphere MQ for Windows Server DVD into the DVD drive.
2. If autorun is installed, the installation process starts.

   Otherwise, double-click the **Setup** icon in the root folder of the DVD to start the installation process.

   The WebSphere MQ Installation Launchpad window is displayed.
3. Click the **WebSphere MQ Installation** option.
4. Click **Launch WebSphere MQ Installer**. Wait until the IBM WebSphere MQ Setup window is displayed with a welcome message.
5. If you have multiple installations on your system, you must choose the installation you want to modify. Do this by selecting the **Maintain or upgrade an existing instance** option and choosing the appropriate instance. If you are upgrading a IBM WebSphere MQ Version 7.0.1 installation (or earlier) to Version 7.1.0, and you already have a Version 7.1.0 or greater installation, you need to select **Install a new instance**. A subsequent panel then allows you to choose the installation you would like to upgrade.
6. Click **Next** to continue. The Program Maintenance panel is displayed.
7. Select **Modify**, then click **Next**.

The Features panel is displayed.

8. Click the + symbol next to a feature to show any dependent features (subfeatures).
9. To change the installation of a feature:
   a. Click the symbol next to the feature name to display a menu.
   b. Select the required option from:
      - Install this feature
      - Install this feature and all its subfeatures (if any)
      - Do not install this feature (remove if already installed)

      The symbol next to the the feature name changes to show the current installation option.
10. When your selections are complete, click **Next**. IBM WebSphere MQ installation begins.

Silently modifying a WebSphere MQ server installation using msiexec:

To silently modify an installation using msiexec, set the ADDLOCAL parameter to include the features you want to add, and set the REMOVE parameter to the features you want to remove.

For example if you use ADDLOCAL="JavaMsg" and REMOVE="" it modifies the installation to include the Java Messaging and Web Services feature.

```
msiexec /i {PRODUCT CODE} /q ADDLOCAL="JavaMsg" REMOVE="" INSTALLATIONNAME="Installation1"
```

The instructions for msiexec begin here: "Advanced installation using msiexec" on page 231

Post installation tasks:

Find out what tasks can be undertaken once IBM WebSphere MQ has been successfully installed. Begin by following the related pages below:

Related concepts:

"Configuring a WebSphere MQ Server"

"Configuring WebSphere MQ accounts" on page 249
The IBM WebSphere MQ service and queue managers check that any users attempting to access queue managers or queue manager resources such as queues, have the permission to access them.

"Using IBM WebSphere MQ Remotely" on page 246

"Using the Help Center" on page 254

Related tasks:

"Configuring WebSphere MQ with the Prepare WebSphere MQ wizard" on page 246

"Using the Default Configuration wizard" on page 253

"Checking for problems after installing" on page 248
These are optional tasks that you can use to check the installation if you believe there was a problem, or to verify installation messages after an unattended (silent) installation for example.

Related reference:

"Using the Welcome to IBM WebSphere MQ Explorer Content view page" on page 253
The Welcome to IBM WebSphere MQ Explorer Content view page points you to any relevant applications, documentation, tutorials, and education. This page is displayed the first time you launch IBM WebSphere MQ Explorer.

Configuring a WebSphere MQ Server: After installing IBM WebSphere MQ, it is necessary to configure it. The configuration described in this topic is for an environment that uses TCP/IP. The configuration procedure is the same for environments that use other communications protocols (for example, SNA, SPX, or NetBIOS). However, not all of the functions and facilities of IBM WebSphere MQ for Windows are available in these environments. The items that are **not** available are:

- IBM WebSphere MQ Postcard

- IBM WebSphere MQ Explorer

If you are setting up IBM WebSphere MQ for use with the Microsoft Cluster Service (MSCS), see Supporting the Microsoft Cluster Service (MSCS) for more information.

Using IBM WebSphere MQ Remotely: If you are connecting to a Windows machine using either Terminal Services or a Remote Desktop Connection and you have problems creating, starting or deleting a queue manager this might be because of the user right **Create global objects**.

The **Create global objects** user right limits the users authorized to create objects in the global namespace. In order for an application to create a global object, it must either be running in the global namespace, or the user under which the application is running must have the **Create global objects** user right applied to it.

When you connect remotely to a Windows machine using either Terminal Services or Remote Desktop Connection, applications run in their own local namespace. If you attempt to create or delete a queue manager using IBM WebSphere MQ Explorer or the **crtmqm** or **dltmqm** command, or to start a queue manager using the **strmqm** command, it results in an authorization failure. This creates a IBM WebSphere MQ FDC with Probe ID XY132002.

Starting a queue manager using the IBM WebSphere MQ Explorer, or using the **amqmdain qmgr start** command works correctly because these commands do not directly start the queue manager. Instead the commands send the request to start the queue manager to a separate process running in the global namespace.

If you need to perform any of these operations on a queue manager when connected remotely to a Windows machine, you must have the **Create global objects** user right. For information on how to assign a user this right, see your operating system documentation.

Administrators have the **Create global objects** user right by default, so if you are an administrator you can create and start queue managers when connected remotely without altering your user rights.

Configuring WebSphere MQ with the Prepare WebSphere MQ wizard:
About this task

The Prepare IBM WebSphere MQ wizard helps you to configure IBM WebSphere MQ files and a user account for your network, and migrate any queue managers and data from a previous installation. You must run the wizard to configure the WebSphere MQ Service before you can start any queue managers.

The Prepare IBM WebSphere MQ wizard window is displayed when IBM WebSphere MQ installation completes. Follow the instructions given by the wizard to configure IBM WebSphere MQ. At any time while the wizard is running you can click **More Information** in the wizard to view online help about the task you are doing.

Note: On Windows systems with UAC enabled, if you do not complete the Prepare WebSphere MQ Wizard directly after WebSphere MQ is installed, or if for any reason your machine is rebooted between completing WebSphere MQ installation and completing the Prepare WebSphere MQ Wizard, you must accept the Windows prompt when it appears to allow the wizard to run as elevated.

Procedure

1. When WebSphere MQ installation completes, the Prepare WebSphere MQ Wizard window is displayed with a welcome message. To continue, click **Next**
2. If you have run the Prepare IBM WebSphere MQ wizard before, this step is skipped. If you have not run the Prepare IBM WebSphere MQ wizard before, the Prepare IBM WebSphere MQ Wizard window displays a progress bar with the following message:

```
Status: Setting up WebSphere MQ Configuration
```

Wait until the progress bar completes.

3. The Prepare IBM WebSphere MQ Wizard window displays a progress bar with the following message:

   ```
   Status: Setting up the WebSphere MQ Service.
   ```

   Wait until the progress bar completes.

4. IBM WebSphere MQ attempts to detect whether you must configure IBM WebSphere MQ for use with Windows Active Directory Server or later domain users. Depending on the results of the detection, IBM WebSphere MQ does one of the following things:

   - If IBM WebSphere MQ detects that you need to configure IBM WebSphere MQ for Windows Active Directory Server or later domain users, the Prepare IBM WebSphere MQ Wizard window displays a message that starts:

     ```
     WebSphere MQ does not have the authority to query information about
     your user account
     ```

     Optionally, to see online help about configuring the domain account, select More Information. When you are finished, close the IBM WebSphere MQ Help Center window to return to the current window.

     Click **Next**, and go to step 5.

   - If you are not installing on a Windows Active Directory Server or later domain server and IBM WebSphere MQ cannot detect whether you need to configure IBM WebSphere MQ for Windows Active Directory Server or later domain users, the Prepare IBM WebSphere MQ Wizard window displays the following message:

     ```
     Are any of the domain controllers in your network running
     Windows 2000 or later domain server?
     ```

     If you select Yes, click **Next**, then go to step 5.

     If you select No, click **Next**, then go to step 9.

     If you select **Don't know**, you cannot continue. Select one of the other options, or click **Cancel** and contact your domain administrator.

   - If IBM WebSphere MQ detects that you do not need to configure IBM WebSphere MQ for Windows Active Directory Server or later domain users, go to step 9.

   Note: At any time, you can click **More Information** to view online help about configuring the domain account, or see “Configuring WebSphere MQ accounts” on page 249. When you are finished, close the IBM WebSphere MQ Help Center window to return to the current window.

5. The Prepare IBM WebSphere MQ Wizard window displays the following message:

   ```
   Do you need to configure WebSphere MQ for users defined on Windows 2000
   or later domain controllers?
   ```

   If you select Yes, click **Next**, then go to step 6.

   If you select No, click **Next**, then go to step 9.

   If you select Don't know, you cannot continue. Select one of the other options, or click **Cancel** and contact your domain administrator.

   Note: At any time, you can click **More Information** to view online help about configuring the domain account, or see “Configuring WebSphere MQ accounts” on page 249. When you are finished, close the IBM WebSphere MQ Help Center window to return to the current window.

6. Give the domain user that you obtained from your domain administrator the right to run as a service.

   a. Click **Start** > **Run...**, type the command **`secpol.msc`** and click **OK**.

   b. Open **Security Settings** > **Local Policies** > **User Rights Assignments**. In the list of policies, right-click **Log on as a service** > **Properties**.

c. Click **Add User or Group...** and type the name of the user you obtained from your domain administrator, and click **Check Names**

d. If prompted by a Windows Security window, type the user name and password of an account user or administrator with sufficient authority, and click **OK** > **Apply** > **OK**. Close the Local Security Policy window.

7. In the next window, enter the Domain and user ID of the domain user account that you obtained from your domain administrator. Either enter the Password for this account, or select the option **This account does not have a password**. Click **Next**.
8. The Prepare IBM WebSphere MQ Wizard window displays a progress bar with the following message:

   ```
   Status: Configuring WebSphere MQ with the special domain user account
   ```

   Wait until the progress bar completes.

   If there are any problems with the domain user account, a further window is displayed. Follow the advice on this window before you continue with this procedure.
9. The Prepare IBM WebSphere MQ Wizard window displays a progress bar with the following message:

   ```
   Status: Starting WebSphere MQ services
   ```

   Wait until the progress bar completes.
10. Next, select the options that you require. The Prepare IBM WebSphere MQ Wizard window displays the following message:

    ```
    You have completed the Prepare WebSphere MQ Wizard
    ```

    Select the options that you require, then click **Finish**. Select one or more from:
    - **Remove the shortcut to this wizard from the desktop**

      This option is available only if you have previously attempted installation, but you canceled the procedure from the Prepare IBM WebSphere MQ wizard and you created a desktop shortcut to this wizard. Select this option to remove the shortcut. You do not need it now that you have completed the Prepare IBM WebSphere MQ wizard.
    - **Launch IBM WebSphere MQ Explorer**

      The IBM WebSphere MQ Explorer allows you to view and administer your IBM WebSphere MQ network.
    - **Launch Notepad to view the release notes**

      The release notes contain information about installing IBM WebSphere MQ and also late-breaking news that is available after the published documentation is produced.
11. Follow the procedure described in “Checking for problems after installing.”

Related information:

User rights required for a WebSphere MQ Windows Service

Checking for problems after installing:

These are optional tasks that you can use to check the installation if you believe there was a problem, or to verify installation messages after an unattended (silent) installation for example.

About this task

Use these steps as a guide to check the following files for messages:

Procedure

1. `MSI`*nnnnn*`.LOG`. This file is in your user `Temp` folder. It is an application log that contains English messages written during installation. The log includes a message indicating whether the installation was successful and complete.

   This file is created if you have set up default logging.
2. If you used the launchpad to install IBM WebSphere MQ, check `MQv7_Install_YYYY-MM-DDTHH-MM-SS.log` in your user `Temp` folder, where:

   YYYY This is the year that you installed WebSphere MQ Version 7.0

   MM This is the month that you installed IBM WebSphere MQ, for example this would be 09 if you installed in September

   DD This is the day that you installed IBM WebSphere MQ

   HH-MM-SS
   This is the time at which IBM WebSphere MQ was installed

   You can get to your user Temp directory by entering the following command at the command prompt:

   ```
   cd %TEMP%
   ```
3. `amqmjpse.txt`. This file is in the IBM WebSphere MQ data files folder (default `c:\Program Files\IBM\WebSphere MQ`). It is an application log that contains English messages written during installation by the Prepare IBM WebSphere MQ wizard.

What to do next

1. Verify your installation, as described in "Verifying a server installation" on page 280

Configuring WebSphere MQ accounts:

The IBM WebSphere MQ service and queue managers check that any users attempting to access queue managers or queue manager resources such as queues, have the permission to access them.

Most networked Windows systems are members of a Windows domain where user accounts, other security principals, and security groups are maintained and managed by a directory service, Active Directory, running on a number of domain controllers. IBM WebSphere MQ checks that only authorized users can access queue managers or queues.

In such networks, IBM WebSphere MQ queue manager processes access the Active Directory information to find the security group membership of any users attempting to use IBM WebSphere MQ resources. The accounts under which IBM WebSphere MQ services run must be authorized to look up such information from the directory. In most Windows domains, local accounts defined at individual Windows servers cannot access directory information, so the IBM WebSphere MQ services must run under a domain account that has the appropriate permission.

If the Windows server is not a member of a Windows domain or the domain has a reduced security or functional level, then the IBM WebSphere MQ services can run under a local account that was created during installation.

Assuming that a domain account is needed, provide the information described in the Information for domain administrator to your domain administrator, and ask for one of the special accounts it describes. When you install the product, towards the end of the installation procedure, in the Prepare IBM WebSphere MQ wizard, you are asked to enter details of this account (domain, user name, and password).

If a domain account is needed and you install IBM WebSphere MQ without a special account (or without entering its details), many or all parts of IBM WebSphere MQ do not work, depending upon the particular user accounts involved. Also, IBM WebSphere MQ connections to queue managers that run under domain accounts on other systems might fail. The account can be changed by running the Prepare IBM WebSphere MQ wizard and specifying the details of the account to be used.

For information about the user rights required to take advantage of the Active Directory support, see Using Active directory (Windows only).

For information about the user rights required to take advantage of the Kerberos authentication support, see Security.

Information for domain administrators:

Use this topic to understand how IBM WebSphere MQ services check the authorization of user accounts attempting to access IBM WebSphere MQ.

The user account must either have an individual IBM WebSphere MQ authorisation set or belong to a local group that has been authorized. A domain account can also be authorized through membership of a domain group included under an authorized local group through a single level of nesting.

The account under which the IBM WebSphere MQ services are run must have the ability to query group memberships of domain accounts and have the authority to administer IBM WebSphere MQ. Without the ability to query group memberships the access checks made by the services fail.

On most Windows domains, with domain controllers running Windows Active Directory, local accounts do not have the required authorization and a special domain user account with the required permissions must be used. The IBM WebSphere MQ installer must be given the userid and password details so that they can be used to configure the IBM WebSphere MQ service after the product is installed.

Typically, this special account has the IBM WebSphere MQ administrator rights through membership of the domain group `DOMAIN\Domain mqm`. The domain group is automatically nested by the installation program under the local mqm group of the system on which IBM WebSphere MQ is being installed.

See "Creating and setting up domain accounts for IBM WebSphere MQ" for instructions on creating a suitable domain account.

Note: If an installer configures IBM WebSphere MQ without a special account, many or all parts of IBM WebSphere MQ do not work, depending upon the particular user accounts involved, as follows:

- An installer currently logged on with a domain user account is not be able to complete the Default Configuration, and the Postcard application does not work.
- IBM WebSphere MQ connections to queue managers running under domain accounts on other systems might fail.
- Typical errors include "AMQ8066: Local mqm group not found" and "AMQ8079: Access was denied when attempting to retrieve group membership information for user 'abc@xyz'".

Creating and setting up domain accounts for IBM WebSphere MQ:

The following information is intended at Domain Administrators. Use this information to create and setup domain accounts for IBM WebSphere MQ.

About this task

Repeat Steps 1 on page 251 and 2 on page 252 for each domain that has user names that will install IBM WebSphere MQ, to create an account for IBM WebSphere MQ on each domain:

Procedure

1. Create a domain group with a special name that is known to IBM WebSphere MQ and give members of this group the authority to query the group membership of any account:
   a. Log on to the domain controller as an account with domain administrator authority.
   b. From the Start menu, open Active Directory Users and Computers.
   c. Find the domain name in the navigation pane, right-click it and select **New Group**.
   d. Type a group name into the **Group name** field.

      Note: The preferred group name is `Domain mqm`. Type it exactly as shown.
      - Calling the group `Domain mqm` modifies the behavior of the "Prepare IBM WebSphere MQ" wizard on a domain workstation or server. It causes the "Prepare IBM WebSphere MQ" wizard automatically to add the group `Domain mqm` to the local `mqm` group on each new installation of IBM WebSphere MQ in the domain.
      - You can install workstations or servers in a domain with no `Domain mqm` global group. If you do so, you must define a group with the same properties as `Domain mqm` group. You must make that group, or the users that are members of it, members of the local `mqm` group wherever IBM WebSphere MQ is installed in a domain. You can place domain users into multiple groups. Create multiple domain groups, each group corresponding to a set of installations that you want to manage separately. Split domain users, according to the installations they manage, into different domain groups. Add each domain group or groups to the local `mqm` group of different IBM WebSphere MQ installations. Only domain users in the domain groups that are members of a specific local `mqm` group can create, administer, and run queue managers for that installation.
      - The domain user that you nominate when installing IBM WebSphere MQ on a workstation or server in a domain must be a member of the `Domain mqm` group, or of an alternative group you defined with same properties as the `Domain mqm` group.
   e. Leave **Global** clicked as the **Group scope**, or change it to **Universal**. Leave **Security** clicked as the **Group type**. Click **OK**.
   f. Follow these steps to assign permissions to the group based on the Windows version of the domain controller:

      On Windows Server 2008 and later versions:
      1) In the Server Manager action bar, click **View** > **Advanced features**.
      2) In the Server Manager navigation tree, click **Users**.
      3) In the Users window, right-click **Domain mqm** > **Properties**.
      4) Click **Security** > **Advanced** > **Add**. Type `Domain mqm` and click **Check names** > **OK**.

         The **Name** field is prefilled with the string, `Domain mqm (`*`domain name`*`\Domain mqm)`.
      5) Click **Properties**. In the **Apply to** list, select **Descendant User Objects**.
      6) From the **Permissions** list, select the **Read group membership** and **Read groupMembershipSAM Allow** check boxes; click **OK** > **Apply** > **OK** > **OK**.

      On Windows 2003 Server:
      1) In the Server Manager action bar, click **View** > **Advanced features** > **Active Directory Users and Computers**.
      2) In the Server Manager navigation tree, search for the domain name. Select the domain name, right-click and select **Properties**.
      3) Click **Security** > **Advanced** > **Add**. Type `Domain mqm` and click **Check names** > **OK**.
      4) Click **Properties**. In the **Apply to** list, select **User Objects**
      5) From the **Permissions** list, select the **Read group membership** and **Read groupMembershipSAM Allow** check boxes; click **OK** > **Apply** > **OK** > **OK**.

      On Windows 2000 Server:

1) In the Server Manager navigation tree, search for the domain name. Select the domain name, right-click and select **Delegate Control Next**.

2) Click **Selected Groups and Users** > **Add...**. Select `Domain mqm` and click **Add** > **OK**.

3) Select `Domain mqm` and click **Next**.

4) Click **Create a custom task to delegate** and click **Next**.

5) Select **Only the following objects in the folder**, and then check `User Objects` in the alphabetical list. Click **Next**.

6) Check **Property-specific**, then select the following options from the list:

   - **Read group membership**
   - **Read groupMembershipSAM**

   Note: The list is in alphabetical order by the second word.

7) Click **OK** to close each window.

2. Create one or more accounts, and add them to the group:

   a. In **Active Directory Users and Computers**, create a user account with a name of your choosing and add it to group `Domain mqm` (or a group that is a member of the local `mqm` group).

   b. Repeat for all the accounts you want to create.

3. Repeat Steps 1 on page 251 and 2 for each domain that has user names that will install IBM WebSphere MQ, to create an account for IBM WebSphere MQ on each domain.

4. Use the accounts to configure each installation of IBM WebSphere MQ:

   a. Either use the same domain user account (as created in Step 1 on page 251) for each installation of IBM WebSphere MQ, or create a separate account for each one, adding each to the `Domain mqm` group (or a group that is a member of the local `mqm` group).

   b. When you have created the account or accounts, give one to each person configuring an installation of IBM WebSphere MQ. They must enter the account details (domain name, user name, and password) into the Prepare IBM WebSphere MQ wizard. Give them the account that exists on the same domain as their installing userid.

   c. When you install IBM WebSphere MQ on any system on the domain, the IBM WebSphere MQ install program detects the existence of the `Domain mqm` group on the LAN, and automatically adds it to the local `mqm` group. (The local `mqm` group is created during installation; all user accounts in it have authority to manage IBM WebSphere MQ). Thus all members of the "`Domain mqm`" group will have authority to manage IBM WebSphere MQ on this system.

   d. However, you do still need to provide a domain user account (as created in Step 1 on page 251) for each installation, and configure IBM WebSphere MQ to use it when making its queries. The account details must be entered into the Prepare IBM WebSphere MQ wizard that runs automatically at the end of installation (the wizard can also be run at any time from the **start** menu).

5. Set the password expiry periods:

   - If you use just one account for all users of IBM WebSphere MQ, consider making the password of the account never expire, otherwise all instances of IBM WebSphere MQ will stop working at the same time when the password expires.
   - If you give each user of IBM WebSphere MQ their own user account you will have more user accounts to create and manage, but only one instance of IBM WebSphere MQ will stop working at a time when the password expires.

   If you set the password to expire, warn the users that they will see a message from IBM WebSphere MQ each time it expires - the message warns that the password has expired, and describes how to reset it.

6. Running IBM WebSphere MQ as a service. If you need to run IBM WebSphere MQ as a service, and then give the domain user (that you obtained from your domain administrator) the right to run as a service, carry out the following procedure:

a. Click **Start > Run...**. Type the command `secpol.msc` and click **OK**.
b. Open **Security Settings > Local Policies > User Rights Assignments**. In the list of policies, right-click **Log on as a service > Properties**.
c. Click **Add User or Group...** Type the name of the user you obtained from your domain administrator, and click **Check Names**
d. If prompted by a Windows Security window, type the user name and password of an account user or administrator with sufficient authority, and click **OK > Apply > OK**. Close the Local Security Policy window.

Note: On Windows Vista and Windows Server 2008 the User Account Control (UAC) is enabled by default.

The UAC feature restricts the actions users can perform on certain operating system facilities, even if they are members of the Administrators group. You must take appropriate steps to overcome this restriction.

Using the Default Configuration wizard:
About this task

You can use the Default Configuration wizard to add the first configured queue manager to this system. This enables you to connect easily with other queue managers in the same IBM WebSphere MQ cluster. You can use the Default Configuration wizard to create, view, or alter your default configuration. You can also use this wizard to alter or display details of an existing queue manager that was created by the default configuration.

For a new installation of IBM WebSphere MQ, creating a default configuration enables you to explore features of IBM WebSphere MQ using the Postcard application, and the IBM WebSphere MQ Explorer.

The Postcard application provides a fast and simple way to verify that your IBM WebSphere MQ installation completed successfully. It uses the default queue manager that is created during the default configuration. If you want to use the Postcard application for verification, and you do not have any existing queue managers, run the Default Configuration wizard first.

If you have migrated existing queue managers, or created any queue managers since installing IBM WebSphere MQ, you might not want to run the Default Configuration wizard. This is because you cannot create the default configuration if other queue managers already exist. If you have previously created any other queue managers on this system and you still want to set up a default configuration, you must delete them before you run the Default Configuration wizard.

Start the Default Configuration wizard by selecting **Create the Default Configuration** on the Welcome to WebSphere MQ Explorer Content view page.

Using the Welcome to IBM WebSphere MQ Explorer Content view page:

The Welcome to IBM WebSphere MQ Explorer Content view page points you to any relevant applications, documentation, tutorials, and education. This page is displayed the first time you launch IBM WebSphere MQ Explorer.

You can use the items in the Welcome to IBM WebSphere MQ Explorer Content view page to explore the facilities in IBM WebSphere MQ. This page is launched the first time the IBM WebSphere MQ Explorer is launched. The Welcome page can be viewed at any time from the Explorer by clicking **IBM WebSphere MQ** in the Navigator view. There are links to the following subjects from this page:

Create the Default Configuration

Allows you to add a configured queue manager to this system for connecting easily with other queue managers in the same IBM WebSphere MQ cluster. You can also use it to alter or display details of an existing queue manager created by the default configuration. This feature is available only using TCP/IP.

Note: If you migrated existing queue managers, or if you have created any queue managers after you installed IBM WebSphere MQ, you might not want to use this facility. This is because you can only set up a default configuration if there are no queue managers already, and you might not want to delete your existing queue managers.

Launch Postcard

Allows you to try out IBM WebSphere MQ messaging quickly and easily. You can send a message either to your own machine or to another named user's machine. It is described in detail in "Verify the installation using the Postcard application" on page 286.

Using the Help Center:

The Help Center gives you access to all task-oriented help, information on the IBM website, and a link to the IBM WebSphere MQ product documentation if you have installed it from the IBM WebSphere MQ Documentation CD.

The IBM WebSphere MQ Help Center can be accessed from the IBM WebSphere MQ Explorer by selecting **Help** > **Help Contents**.

Converting a trial license on UNIX, Linux, and Windows

Convert a trial license to a full license without reinstalling IBM WebSphere MQ.

When the trial license expires, the "count-down" displayed by the **strmqm** command informs you the license has expired, and the command does not run.

Before you begin

1. IBM WebSphere MQ is installed with a trial license.
2. You have access to the installation media of a fully licensed copy of IBM WebSphere MQ.

About this task

Run the **setmqprd** command to convert a trial license to a full license.

If you do not want to apply a full license to your trial copy of IBM WebSphere MQ, you can uninstall it at any time.

Procedure

1. Obtain the full license from the fully licensed installation media.

   The full license file is **amqpcert.lic**. On UNIX and Linux, it is in the `/`*`MediaRoot`*`/licenses` directory on the installation media. On Windows it is in the `\`*`MediaRoot`*`\licenses` directory on the installation media. It is installed into the `bin` directory on the IBM WebSphere MQ installation path.
2. Run the **setmqprd** command from the installation that you are upgrading:

   ```
   $MQ_INSTALLATION_PATH/bin/setmqprd /MediaRoot/licenses/amqpcert.lic
   ```

Related information:

setmqprd

Displaying messages in your national language on UNIX and Linux systems

To display messages from a different national language message catalog, you must install the appropriate catalog and set the **LANG** environment variable.

About this task

Non-AIX platforms
Messages in U.S. English are automatically installed with WebSphere MQ.

AIX Messages in the language specified by the locale selected on your machine at install time are installed by default.

To find out which language is currently in use, run the **locale** command.

If this returns a language which is not one of the national languages provided by WebSphere MQ, you must select a national language, otherwise you will not get a message catalog installed on your system.

Message catalogs for all languages are installed in *MQ_INSTALLATION_PATH*/msg/*language identifier*, where *language identifier* is one of the identifiers in Table 44.

If you require messages in a different language, perform the following steps:

Procedure

1. Install the appropriate message catalog (see "Choosing what to install" on page 159).
2. To select messages in a different language, ensure the **LANG** environment variable is set to the identifier for the language you want to install:

Table 44. Language identifiers

Identifier	Language
cs_CZ	Czech
de_DE	German
es_ES	Spanish
fr_FR	French
hu_HU	Hungarian
it_IT	Italian
ja_JP	Japanese
ko_KR	Korean
pl_PL	Polish
pt_BR	Brazilian Portuguese
ru_RU	Russian
zh_CN	Simplified Chinese
zh_TW	Traditional Chinese

AIX has some additional message catalogs:

Table 45. AIX specific language identifiers

Identifier	Language
Ja_JP	Japanese
Zh_CN	Simplified Chinese
Zh_TW	Traditional Chinese

Displaying messages in your national language on Windows systems

Windows

To display messages from a different national language message catalog, you must either set the **MQS_FORCE_NTLANGID** environment variable, or change a Regional setting.

About this task

Messages in U.S. English are automatically installed with IBM WebSphere MQ

Messages in the national languages that IBM WebSphere MQ supports are automatically installed. Messages are displayed in the national language, based on the following order:

1. The value of the **MQS_FORCE_NTLANGID** environment variable, if set.
2. The Region Format of the user that is displaying the message, if the language specified by the Region Format is supported by IBM WebSphere MQ.
3. The Administrative system locale if the language specified by the system locale is supported by IBM WebSphere MQ.
4. US English, if no other supported language can be determined.

Note: The queue manager is usually launched by a service on the machine, and hence is running under its own user account (for example MUSR_MQADMIN) or a specific domain account provided during install time. See Security on Windows for more information.

If you require messages in a language other than the one associated with the Region Format of a user account, perform the following steps:

Procedure

1. Globally set the **MQS_FORCE_NTLANGID** environment variable, to the language identifier of the desired language, for messages displayed by the queue manager. You should set the **MQS_FORCE_NTLANGID** system wide. Otherwise, every user displaying messages needs to have the environment variable set individually.

   The language identifier values, represented in hexadecimal notation, are listed in the following Microsoft document: Language Identifier Constants and Strings
2. Reboot machines where queue managers are running as a service, for the environment variable to take effect.

Installing a IBM WebSphere MQ client

After preparing your system for installation, you can install a IBM WebSphere MQ client by following the appropriate instructions for your platform. After installation, you might want to verify your installation to check that installation has been successful.

Before you start the installation procedure, make sure that you have prepared your system as described in Preparing the system

To begin the installing procedure, select the appropriate platform:

- "Installing a IBM WebSphere MQ client on AIX systems"
- "Installing a IBM WebSphere MQ client on HP-UX systems" on page 260
- "Installing WebSphere MQ client on Linux" on page 262
- "Installing a IBM WebSphere MQ client on Solaris" on page 264
- "Installing a IBM WebSphere MQ client on Windows systems" on page 265

Installing IBM WebSphere MQ clients and servers on the same system

To install a IBM WebSphere MQ client on a system that is already running a IBM WebSphere MQ server, use the appropriate Server DVD. Use a Client DVD to install a IBM WebSphere MQ client only on a system that is not running a IBM WebSphere MQ server.

If you install a IBM WebSphere MQ client from a Client DVD and later decide to install the IBM WebSphere MQ server on the same system, you must first remove all the client components from the system. Then use the appropriate Server DVD to install both the server and client components. You cannot install a IBM WebSphere MQ server on a system that already has client components installed from a Client DVD.

Remember that even if your client and server are installed on the same system, you must still define the MQI channel between them. See Defining MQI channels for details.

Installing a IBM WebSphere MQ client on AIX systems

You can interactively install the IBM WebSphere MQ client for AIX using **smit**.

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

About this task

IBM WebSphere MQ is supplied as a set of filesets that are installed using the standard AIX installation tools. The procedure uses the System Management Interface Tool (**smit**), but you can choose to use **`installp`**, **`geninstall`** or the web-based System Manager. You can select which components you want to install. The components and filesets are listed in "Choosing what to install" on page 159. You must install at least the Runtime and Client components.

This procedure installs IBM WebSphere MQ into the default location. If you want to install to a non-default location, you must use **`installp`**, see "Non-interactive installation of IBM WebSphere MQ client for AIX" on page 258.

Procedure

1. Log in as root, or switch to the superuser using the **`su`** command.
2. Make your current directory the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. Select the required **smit** window using the following sequence:
   ```
   Software Installation and Maintenance
    Install and Update Software
     Install and Update from ALL Available Software
   ```
4. Click **List** to display the input device or directory for the software and select the location that contains the installation images.

5. Select the **SOFTWARE to install** field to obtain a list of available filesets, and select the filesets you want to install. Ensure that you include the appropriate message catalog if you require messages in a language different from the language specified by the locale specified on your system. Enter **ALL** to install all applicable filesets.
6. Change **Preview new LICENSE agreements?** to **yes** and press Enter to view the license agreements.
7. If you have a previous version of the product on your system, change the **Automatically install requisite software** to **no**.
8. Change **ACCEPT new license agreements?** to **yes** and press Enter to accept the license agreements.
9. Change **Preview new LICENSE agreements?** to **no** and press Enter to install IBM WebSphere MQ.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- For instructions on how to verify your installation, see “Verifying a client installation” on page 290.

Non-interactive installation of IBM WebSphere MQ client for AIX:

Silently install IBM WebSphere MQ client from the command line using the AIX **installp** command.

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in “Preparing the system” on page 196.

Note: Installation to a non-default location is *not* supported on systems that have the AIX Trusted Computing Base (TCB) enabled.

About this task

You can use this method to install to a non-default location, and can select which components you want to install. The components and filesets are listed in “Choosing what to install” on page 159. You must install at least the Runtime and Client components.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Make your current directory the location of the installation file. The location might be the mount point of the CD, a network location, or a local file system directory.
3. Install the product in one of the following ways:
   - Install the whole product in the default location:

     ```
     installp -acgXYd . all
     ```
   - Install selected filesets in the default location:

     ```
     installp -acgXYd . list of file sets
     ```
   - Install the whole product in a non-default location using the -R flag:

     ```
     installp -R MQ_INSTALLATION_PATH -acgXYd . all
     ```

- Install selected filesets in a non-default location using the -R flag:

  ```
  installp -R MQ_INSTALLATION_PATH -acgXYd . list of file sets
  ```

  The directory specified with the -R flag is an AIX User Specified Install Location (USIL). WebSphere MQ is installed underneath the directory specified. For example, if `-R /USIL1` is specified, the WebSphere MQ product files are located in `/USIL1/usr/mqm`.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Installing IBM WebSphere MQ client on HP Integrity NonStop Server systems

Installing a IBM WebSphere MQ client on a HP Integrity NonStop Server system.

Before you begin

Before you start the installation procedure, make sure that you complete the necessary steps that are outlined in "Setting up the user and group on HP Integrity NonStop Server" on page 199.

About this task

After preparing your system for installation, install the IBM WebSphere MQ client for HP Integrity NonStop Server by following the instructions. After installation, you might want to verify your installation to check that it installed successfully. There are three steps to the installation:

1. Downloading the installation package.
2. Running the installer.
3. Setting the environment.

Procedure

1. Log in to the OSS user ID that owns the installation. The OSS user ID must have MQM as its primary group.
2. Download the installation package file. Ensure that you use "binary mode" when you download the installation package file to OSS. Any corruption in the file causes the self-extracting archive to fail to run. After you have downloaded the package file, ensure that it has read and execute permissions for the user ID that is installing the package.
3. Set the *_RLD_FIRST_LIB_PATH* variable to `<install path>/opt/mqm/bin`
4. Optional: Make your current directory the location of the installation file.
5. Type the following command to start the interactive installation procedure:

   ```
   ./<name of package file> -i <OSS install_root> -g <Guardian install_root>
   ```

   where

   `<name of package file>` is the name of the installation package.

   `<OSS install_root>` is the OSS root directory of the new installation.

   `<Guardian install_root>` is the Guardian subvolume for the new installation.

Both -i and -g options are mandatory.

- -i specifies the new or empty OSS directory that contains the opt/mqm and var/mqm directories of the installation.
- -g specifies the subvolume into which the Guardian components of the IBM WebSphere MQ client on a HP Integrity NonStop Server are installed. The Guardian subvolume can be specified in either OSS-form or Guardian-form and can be abbreviated. The Guardian subvolume specification is not case sensitive. The following are examples of valid Guardian subvolume specifications:
  - /G/vol/subvol
  - vol/subvol
  - \$VOL.SUBVOL
  - vol.subvol

6. Optional: For OSS, set your environment by installing the binaries into your path. To do this, type the following command:

   ```
   export PATH=$PATH:<OSS_install_root>/opt/mqm/bin
   ```

   where <OSS_install_root> is the OSS root directory of the new installation.

Example

To install the IBM WebSphere MQ client for HP Integrity NonStop Server from package mat1.run, type the following command:

```
 ./mat1.run -i ~install/mq75client -g /G/data04/mqm
```

The command installs the OSS components into new opt/mqm and var/mqm directories in ~install/mq75client. It installs the Guardian components into /G/data04/mqm.

What to do next

For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Installing a IBM WebSphere MQ client on HP-UX systems

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

About this task

This topic describes the installation of a client, using the **swinstall** program to select which components you want to install. The components and are listed in "Choosing what to install" on page 159; you must install at least the Runtime and Client components.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Make your current directory the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. Accept the license by running the mqlicense script:

   ```
   ./mqlicense.sh
   ```

   The license is displayed. If you accept the license, you can continue the installation.
4. Type the following command to start the interactive installation procedure:

   ```
   swinstall -s installation_file
   ```

installation_file is the absolute path to the installation file. The path must begin with a / and end with the name of the installation file. The installation file has the extension `.v11`.

If the files on your DVD are in uppercase with a ";1" suffix, use this name for the depot.

5. In the resulting menu screen, select **MQSERIES**.
   a. If you do not want to install all IBM WebSphere MQ components, open **MQSERIES**
      1) Mark the components you want to install. The installer resolves dependencies automatically.
      2) Review the information displayed by the installer.
6. Optional: To install IBM WebSphere MQ to a non-default location, select **Actions > Change Product Location**.

   For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location.

   The installation path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.
7. Select **Actions > Install**. The log file tells you if there are any problems that need fixing.
8. Fix any problems, and click **OK** to install. You are informed when the installation has finished.
9. If this installation is not the first installation on the system, you must enter the following command:

```
swconfig -x allow_multiple_versions=true MQSERIES,l=MQ_INSTALLATION_PATH
```

   where *MQ_INSTALLATION_PATH* is the path where you have just installed IBM WebSphere MQ. If you do not enter this command, the **swlist** command reports the installation as installed instead of configured. You must not use IBM WebSphere MQ unless the installation is configured.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

```
MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Non-interactive installation of IBM WebSphere MQ client on HP-UX:

You can perform a non-interactive installation of the IBM WebSphere MQ client using the **swinstall** command. A non-interactive installation is also known as a silent, or unattended installation.

Before you begin

Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.

About this task

This topic describes the non-interactive installation of a client, using the **swinstall** program to select which components you want to install. The components and are listed in "Choosing what to install" on page 159; you must install at least the Runtime and client components.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Make your current directory the location of the installation file. The location might be the mount point of the CD, a network location, or a local file system directory.
3. Accept the WebSphere MQ license agreement without an interactive prompt by entering the following command:

   ```
   ./mqlicense.sh -accept
   ```

4. Install WebSphere MQ using the **swinstall** command:
   a. If this installation is not the first installation on the system, you must add `-x allow_multiple_versions=true` to the **swinstall** command.
   b. Add the names of the components to install as parameters of the **swinstall** command. The installer automatically resolves any dependencies.
   c. Optional: Identify the installation location by adding `,l=`*`MQ_INSTALLATION_PATH`* as a parameter of the **swinstall** command. For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location. The installation path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.

   For example, to install all IBM WebSphere MQ components, in a non-default location, as the first installation, enter the following command:

   ```
   swinstall -s /installation_file.v11 MQSERIES,l=/opt/customLocation
   ```

   To perform a partial installation, providing a list of components, in the default location, as the second installation, enter the following command:

   ```
   swinstall -s /installation_file.v11
   MQSERIES.MQM-RUNTIME MQSERIES.MQM-BASE MQSERIES.MQM-CL-HPUX -x allow_multiple_versions=true
   ```

   `/installation_file.v11` is the absolute path to the installation file. The path must begin with a `/` and end with the name of the installation file. The installation file has the extension `.v11`.
5. If this installation is not the first installation on the system, you must enter the following command:

   ```
   swconfig -x allow_multiple_versions=true MQSERIES,l=MQ_INSTALLATION_PATH
   ```

   where *`MQ_INSTALLATION_PATH`* is the path where you have just installed IBM WebSphere MQ. If you do not enter this command, the **swlist** command reports the installation as installed instead of configured. You must not use IBM WebSphere MQ unless the installation is configured.

What to do next

For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Installing WebSphere MQ client on Linux

Installing a WebSphere MQ client on a 32 bit or 64 bit Linux system.

Before you begin

- Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.
- If this installation is not the first installation on the system, you must ensure that you have write access to `/var/tmp`.

About this task

This task describes the installation of the client, using the RPM Package Manager installer to select which components you want to install. You must install at least the Runtime and Client components. The components are listed in "Choosing what to install" on page 159.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Make your current directory the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. Run the mqlicense.sh script. If you want to view a text-only version of license, which can be read by a screen-reader, type:

   ```
   ./mqlicense.sh -text_only
   ```

   The license is displayed.

   If want to accept the license without it being displayed, you can run the mqlicense.sh script with the -accept option.

   ```
   ./mqlicense.sh -accept
   ```

   You must accept the license agreement before you can proceed with the installation.
4. If this installation is not the first installation on the system, you must run **crtmqpkg** to create a unique set of packages to install on the system:
   1. Enter the following command:

      ```
      crtmqpkg suffix
      ```

      where *suffix* is a name of your choosing, that will uniquely identify the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.
   2. Set your current directory to the location specified when the **crtmqpkg** command completes. This directory is a sub-directory of /var/tmp/mq_rpms, in which the unique set of packages is created. The packages have the *suffix* value contained within the filename.
5. Install IBM WebSphere MQ. The minimum components you must install are the MQSeriesRuntime and the MQSeriesClient.
   - To install to the default location, /opt/mqm, use the **rpm -ivh** command to install each component that you require.

     For example, to install all components to the default location use the following command:

     ```
     rpm -ivh MQSeries*.rpm
     ```
   - To install to a non-default location use the **rpm --prefix** option. For each installation, all of the IBM WebSphere MQ components that you require must be installed in the same location.

     The installation path specified must either be an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces.

     For example, to install the runtime and server components to /opt/customLocation on a 32-bit Linux system:

     ```
     rpm --prefix /opt/customLocation -ivh MQSeriesRuntime-7.5.0-0.i386.rpm
     MQSeriesClient-7.5.0-0.i386.rpm
     ```

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.

- For instructions on how to verify your installation, see "Verifying a client installation" on page 290

Installing a IBM WebSphere MQ client on Solaris

Before you begin

- Before you start the installation procedure, make sure that you have completed the necessary steps outlined in "Preparing the system" on page 196.
- This procedure is for the installation of a standard IBM WebSphere MQ client, from the client DVD. If you are using a server DVD, follow the steps in "Installing IBM WebSphere MQ server on Solaris" on page 224, and select the appropriate client components in step 8.

About this task

This task describes the installation of the IBM WebSphere MQ for Solaris client, using the **pkgadd** program. You can choose which components you want to install. The components (or file sets) are listed in "Choosing what to install" on page 159; you must install at least the Client component.

Note: If you are installing on the Solaris 11 operating system, ensure that the IPS package (package/svr4) that supports **pkgadd** and equivalent utilities is installed.

Procedure

1. Log in as root, or switch to the superuser using the **su** command.
2. Make your current directory the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. Run the `mqlicense.sh` script to accept the license:
   ```
    ./mqlicense.sh
   ```
   If you want to view a text-only version of the license, which can be read by a screen-reader, type:
   ```
   ./mqlicense.sh -text_only
   ```
   The license is displayed. Follow the instructions to accept the license. If you accept the license, the installation continues. If you do not accept the license, you cannot continue the installation process.
4. If this installation is not the first installation on the system, you must run **crtmqpkg** to create a unique set of packages to install on the system:
   a. Enter the following command:
   ```
   ./crtmqpkg suffix
   ```
   where *suffix* is a name of your choosing, that will uniquely identify the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.

   b. Set your current directory to the location specified when the **crtmqpkg** command completes. This directory is a sub-directory of `/var/spool`, in which the unique set of packages is created. The packages have the *suffix* value contained within the filename.
5. Start the installation process:
   - If the installation is the first installation on the system, enter the following command to start the installation process:
     ```
     pkgadd -d ./mqclient.img
     ```
   - If the installation is not the first installation on the system, enter the following command to start the installation process:
     ```
     pkgadd mqm-suffix
     ```
     where *suffix* is the suffix chosen in the previous step.
6. You are presented with a list of the packages that are available. Enter the number of the `mqm` package.

7. You are prompted to choose a location for installation.
   - To install to the default location, enter `y`.
   - To install to a non-default directory, enter `n`. Then enter the required installation path, and confirm your choice.
8. You receive a number of messages, after which a list of components is displayed. Enter the numbers of the components that you require separated by spaces or commas.
9. If the path chosen in step 7 does not exist, you are asked if you want to create it. You must enter `y` to proceed.
10. Answer any questions appropriately for your system.
11. A message tells you when installation is complete. Enter `q` to exit the **pkgadd** program.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **setmqenv** or **crtmqenv** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Installing a IBM WebSphere MQ client on Windows systems

This topic describes how to install IBM WebSphere MQ client on Windows systems. This procedure can be used for installing a first or a subsequent installation.

Before you begin

To install a IBM WebSphere MQ client, you must be logged on to Windows as an administrator.

About this task

Follow these instructions to perform an interactive compact, typical, or custom installation of IBM WebSphere MQ. To view all the installation types and the features that are installed with each option consult Table 32 on page 182.

Note: To install the extended transactional client use the server installation image.

Procedure

1. Access the IBM WebSphere MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `setup.exe` in the Windows directory of the IBM WebSphere MQ installation image.
   - From a DVD, this location might be:

     ```
     E:\Windows\setup.exe
     ```
   - From a network location, this location might be:

     ```
     m:\instmqs\Windows\setup.exe
     ```
   - From a local file system directory, this location might be:

     ```
     C:\instmqs\Windows\setup.exe
     ```
3. Double-click the **Setup** icon to start the installation process. It is possible to run either by:

- Running `setup.exe` from the command prompt. Or
- Double-clicking `setup.exe` from Windows Explorer.

If you are installing on a Windows system with UAC enabled, accept the Windows prompt to allow the launchpad to run as elevated. During installation, you might also see Open File - Security Warning dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the installation to continue.

The IBM WebSphere MQ Installation window is displayed.

4. Continue to follow the instructions as shown on screen.

Results

A new sample IBM WebSphere MQ MQI client configuration file is created in the IBM WebSphere MQ installation directory (for example `C:\Program Files\IBM\WebSphere MQ\`, by the IBM WebSphere MQ MQI client package, during installation, but only if this file does not exist. This file contains the `ClientExitPath` stanza. An example `mqclient.ini` file is shown in Configuring a client using a configuration file.

Note:

If you are using a common configuration file for multiple clients, either in the IBM WebSphere MQ installation directory or in another location using the MQCLNTCF environment variable, you must grant read access to all user identifiers under which the IBM WebSphere MQ client applications run. If the file cannot be read, the failure is traced and the search logic continues as if the file had not existed.

What to do next

- If you have chosen this installation to be the primary installation on the system, you must now set it as the primary installation. Enter the following command at the command prompt:

  ```
  MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

  You can have only one primary installation on a system. If there is already a primary installation on the system, you must unset it before you can set another installation as the primary installation. For more information, see Changing the primary installation .
- You might want to set up the environment to work with this installation. You can use the **`setmqenv`** or **`crtmqenv`** command to set various environment variables for a particular installation of IBM WebSphere MQ. For more information, see setmqenv and crtmqenv.
- For instructions on how to verify your installation, see "Verifying a client installation" on page 290.

Related concepts:

"Modifying the client installation on Windows using Add/Remove Programs" on page 276
On some versions of Windows, you can modify an installation by using Add/Remove Programs.

Related tasks:

"Advanced installation using msiexec"

"Using the MQParms command" on page 272

Advanced installation using msiexec:

About this task

WebSphere MQ on Windows uses the MSI technology to install software. MSI provides both an interactive installation and a non interactive installation. An interactive installation displays panels and ask questions.

The **msiexec** command uses parameters to give MSI some or all of the information that can also be specified through panels during an interactive installation. This means that a user can create a reusable automated or semi-automated installation configuration. Parameters can be given through the command line, a transform file, a response file, or a combination of the three.

Procedure

To install using msiexec, at the command line, enter the **msiexec** command in the following format:

```
msiexec parameters [USEINI="response-file"] [TRANSFORMS="transform_file"]
```

Where:

parameters
are either command-line parameters preceded by a / character, or property=value pairs (if using both forms of parameter always put the command-line parameters first). For further information, see "Specifying command line parameters with msiexec."

For an unattended installation, you must include the /q or /qn parameter in the command line. Without this parameter, the installation is interactive.

Note: You must include the **/i** parameter and the file location of the IBM WebSphere MQ installer package.

response-file
is the full path and file name of the file that contains the [Response] stanza and the required property=value pairs, for example `C:\MyResponseFile.ini`. An example response file, `Response.ini`, is supplied with IBM WebSphere MQ. This file contains default installation parameters. For further information, see "Using a response file with msiexec" on page 269.

transform_file
is the full path and file name of a transform file. For further information, see "Using transforms with msiexec" on page 271 and "Multiple installation using MSI Instance ID" on page 237.

Note: For a silent installation to succeed, the AGREETOLICENSE=?YES? property must be defined either on the command line or in the response file.

Results

After the command has been entered, the command prompt immediately reappears. IBM WebSphere MQ is installing as a background process. If you have entered parameters to produce a log, check this file to see how the installation is progressing. If the installation completes successfully, you see the message `Installation operation completed successfully` in the log file.

Specifying command line parameters with msiexec:

About this task

The **msiexec** command can accept two types of parameters on the command line, as follows:

- Standard command line parameters, preceded by a / character.

  For a table of the **msiexec** command line parameters, see the MSDN Command-Line Options web page.
- Property=value pair parameters on the command line. All the parameters available for use in a response file can be used on the command line, for a list of these, see Table 47 on page 269. In addition there are some extra property=value pair parameters that are only for use on the command line, for details see Table 46 on page 268.

  When using the property=value pair parameters note that:
  - Property strings must be in uppercase.

- Value strings are not case-sensitive, except for feature names. You can enclose value strings in double quotation marks. If a value string includes a blank, enclose the blank value string in double quotation marks.
- For a property that can take more than one value, use the format:

  ```
  ADDLOCAL="Server,Client"
  ```

When using property=value pair and command line parameters with the **msiexec** command, enter command line parameters first.

If a parameter is specified both on the command line and in a response file, the setting on the command line takes precedence.

Example

A typical example of an **msiexec** command is:

```
msiexec /i "path\MSI\IBM WebSphere MQ.msi" /l*v c:\install.log
/q TRANSFORMS="1033.mst" AGREETOLICENSE="yes" ADDLOCAL="Client"
```

A typical example of an **msiexec** command when you are installing a second copy of IBM WebSphere MQ Version 7.1 is:

```
msiexec /i "path\MSI\IBM WebSphere MQ.msi" /l*v c:\install.log
/q TRANSFORMS=":InstanceId2.mst;1033.mst" AGREETOLICENSE="yes"
ADDLOCAL="Client" MSINEWINSTANCE=1
```

The following table shows the parameters which can only be provided on the command line and not in a response file.

Table 46. msiexec property=value parameters

Property	Values	Meaning
USEINI	*path\file_name*	Use the specified response file. See "Using a response file with msiexec" on page 269
SAVEINI	*path\file_name*	Generate a response file during installation. The file contains those parameters selected for this installation that a user might make during an interactive installation.
ONLYINI	1 | yes | ""	1, yes or any value other than null. End the installation before updating the target system, but after generating a response file, if this is specified. "". Continue the installation and update the target system (the default).
TRANSFORMS	:InstanceId*x*.mst | *path\file_name* | :InstanceId*x*.mst;*path\file_name*	The :InstanceId*x*.mst value is only required for a subsequent installation of IBM WebSphere MQ Version 7.1 or greater. The *path\file_name* specifies what transform (.mst) files must be applied to the product. For example, `"1033.mst"` specifies the supplied U.S. English transform file.
MSINEWINSTANCE	1	This property is only required for subsequent installations of IBM WebSphere MQ Version 7.1 or greater.
REMOVEFEATURES	yes	Required with value "yes" for a silent installation, otherwise ignored. Allows obsolete features, no longer part of IBM WebSphere MQ, to be deleted.

Using a response file with msiexec:

About this task

You can use the **msiexec** command with a parameter which specifies additional properties are defined in a response file. You can combine the msiexec command-line parameters described in “Specifying command line parameters with msiexec” on page 267.

A response file is an ASCII text file, with a format like a Windows .ini file, that contains the stanza [Response]. The [Response] stanza contains some or all the parameters that would normally be specified as part of an interactive installation. The parameters are given in a property=value pair format. Any other stanzas in the response file are ignored by **msiexec**. An example response file, Response.ini, is supplied with IBM WebSphere MQ. It contains the default installation parameters.

Procedure

A typical example of an msiexec command is: `msiexec /i "path\MSI\IBM WebSphere MQ.msi" /l*v c:\install.log    TRANSFORMS="1033.mst" USEINI="C:\MQ\Responsefile"`

If a parameter is specified both on the command line and in a response file, the setting on the command line takes precedence. All the parameters available for use in a response file can also be used on the command line, for a list of these see Table 47.

In the response file, all text is in English, and comments begin with a **;** character.

For information about creating a response file, see “Creating a response file” on page 238.

Example

An example of a typical response file:

```
[Response]
PGMFOLDER="c:\mqm"
DATFOLDER="c:\mqm\data"
AGREETOLICENSE="yes"
ADDLOCAL="Client"
REMOVE="Toolkit"
```

Table 47. Response file parameters

Property	Values	Meaning
PGMFOLDER	*path*	Folder for the IBM WebSphere MQ program files. For example, c:\mqm.
DATFOLDER	*path*	Folder for the IBM WebSphere MQ data files. For example, c:\mqm\data.
USERCHOICE	0|no	If the command line or response file specifies parameters to install features, a dialog can be displayed to prompt you to accept the preselected options, or review and possibly change them. 0 or no. Suppresses display of the dialog. Anything else. Dialog is displayed and you can amend the options. Not used for a silent installation.
AGREETOLICENSE	yes	Accept the terms of the license. Set to yes before a silent installation. If the installation is not silent, this parameter is ignored.

Table 47. Response file parameters (continued)

Property	Values	Meaning
ADDLOCAL	*feature, feature, All|””*	A comma-separated list of features to install locally. For a list of valid feature names, see “IBM WebSphere MQ features for Windows systems” on page 173. All installs all features "" installs the typical features. If you do not want a feature use REMOVE="*feature*" **Note:** If this is a new installation the typical features (Client, Java, .NET Messaging, and Development Toolkit) are installed by default irrespective of the feature list provided in the ADDLOCAL property. If you do not want a feature use REMOVE="feature"
REMOVE	*feature, feature,* |All|""	A comma-separated list of features to remove. For a list of valid feature names, see “IBM WebSphere MQ features for Windows systems” on page 173. All uninstalls all features "" uninstalls no features (the default).
INSTALLATIONDESC	?Description of installation?	Sets the installation description from the command line. Subject to the documented installation description length limitations
INSTALLATIONNAME	[INSTALLATION0,]?Name?	Sets the installation name from the command line. Subject to the documented installation name character and length limitations. **Note:** Supply INSTALLATION0,Name only when upgrading from pre-IBM WebSphere MQ Version 7.1.
MAKEPRIMARY	0|1|""	Makes the installation primary, if possible, or removes the primary flag. 1 = Make primary, 0 = Make non-primary, - use default algorithm **Note:** This option is ignored if a pre-Version 7.1 IBM WebSphere MQ is installed, or if another Version 7.1 or greater installation is present and set as the primary.

Related tasks:

“Using the MQParms command” on page 272

Related reference:

“Using transforms with msiexec” on page 271

Multiple installation using MSI Instance ID:

This topic describes how to choose the MSI instance ID you require for non-interactive multiple installations.

About this task

In order to support non-interactive multiple installations, you need to find out whether the instance ID you want to use is already in use or not and choose the appropriate one. For each installation media (for example, each 7.5 client and 7.5 server), Instance ID 1 is the default ID which is used for single installations. If you want to install alongside Instance ID 1 you need to specify which instance you want to use. If you have already installed instance 1, 2, and 3 then you need to find out what the next available instance is, for instance, Instance ID 4. Similarly, if instance 2 has been removed, you need to find out that there is a gap that can be reused. You can find out which Instance ID is currently in use using the **dspmqinst** command.

Procedure

1. Type **dspmqinst** to find a free MSI Instance in the media being installed by reviewing the MSIMedia and MSIInstanceId values for the versions already installed. For example:

   ```
   InstName:      Installation1
   InstDesc:
   Identifier:    2
   InstPath:      C:\mq\install1
   Version:       7.5.0.0
   Primary:       Yes
   State:         Available
   MSIProdCode:   {0730749B-080D-4A2E-B63D-85CF09AE0EF0}
   MSIMedia:      7.5 Server
   MSIInstanceId: 1
   ```

2. If MSI Instance ID 1 is in use and you want to use MSI Instance ID 2, the following parameters must be added to the msiexec call:

   ```
   MSINEWINSTANCE=1 TRANSFORMS=:InstanceId2.mst
   ```

What to do next

For multiple installations, the **INSTALLATIONNAME** or **PGMFOLDER** must be supplied as an additional parameter on any non-interactive installation command. Supplying the **INSTALLATIONNAME** or **PGMFOLDER** ensures that you do not work with the wrong installation in case you omit or incorrectly specify the **TRANSFORMS** parameter.

Using transforms with msiexec:
MSI can use transforms to modify an installation. During IBM WebSphere MQ installation, transforms can be used to support different national languages. IBM WebSphere MQ is supplied with transform files in the \MSI folder of the client image. These files are also embedded in the IBM WebSphere MQ Windows installer package, IBM WebSphere MQ.msi.

On the **msiexec** command line, you can specify the required language by using the TRANSFORMS property in a property=value pair. For example:

```
TRANSFORMS="1033.mst"
```

You can also specify the full path and file name of the transform file. Again, the quotation marks surrounding the value are optional. For example:

```
TRANSFORMS="D:\Msi\1033.mst"
```

Table 48 on page 272 shows the locale identifier, language, and the transform file name to use in the **msiexec** command line.

You might need to merge transforms to install multiple installations of the same version, for example:

```
TRANSFORMS=":InstanceId2.mst;D:\Msi\1033.mst"
```

You can also specify the required language by using the MQLANGUAGE property with the **MQParms** command. For information about the msiexec property=value parameters, see "MQParms parameter file" on page 274.

Parameters

Table 48. Supplied transform files for various language support. This table shows the supplied transform files, the resulting language, and the numeric value to use in the **msiexec** command line.

Language	Transform File name	Value
U.S. English	1033.mst	1033
German	1031.mst	1031
French	1036.mst	1036
Spanish	1034.mst	1034
Italian	1040.mst	1040
Brazilian Portuguese	1046.mst	1046
Japanese	1041.mst	1041
Korean	1042.mst	1042
Simplified Chinese	2052.mst	2052
Traditional Chinese	1028.mst	1028
Czech	1029.mst	1029
Russian	1049.mst	1049
Hungarian	1038.mst	1038
Polish	1045.mst	1045

Creating a response file:

A response file is used with **msiexec**. You can create it in three ways.

About this task

A response file is used with the **msiexec** command, for further information see "Using a response file with msiexec" on page 233.

Procedure

There are three ways to create a response file for installation:

- Copy and edit the file `Response.ini` that is supplied on the IBM WebSphere MQ Windows Server CD, using an ASCII file editor.
- Create your own response file using an ASCII file editor.
- Use the **msiexec** command with the **SAVEINI** (and optionally, the **ONLYINI**) command line parameters to generate a response file that contains the same installation options. See Table 38 on page 233.

Example

A typical example of using **msiexec** with the **SAVEINI** parameter is here:

```
msiexec /i "path\IBM WebSphere MQ.msi" /q SAVEINI="response_file"
TRANSFORMS="1033.mst" AGREETOLICENSE="yes"
```

Using the MQParms command:

Before you begin

You can use the `MQParms` command to invoke installation or uninstallation. This command can use parameters on a command line, or those specified in a parameter file. The parameter file is an ASCII text file that contains the parameter values that you want to set for the installation. The MQParms command takes the specified parameters and generates the corresponding msiexec command line.

This means that you can save all the parameters that you want to use with the msiexec command in a single file.

If you are running IBM WebSphere MQ on Windows systems with User Account Control (UAC) enabled, you must invoke the installation with elevated privileges. If you are using the Command prompt or IBM WebSphere MQ Explorer elevate privileges by using a right-click to start the program and selecting **Run as administrator**. If you try to run the MQParms program without using elevated privileges, the installation fails with an error of AMQ4353 in the installation log.

For silent operations, this must include the **/q** or **/qn** parameter, either on the command line, or in the [MSI] stanza of the parameter file. You must also set the AGREETOLICENSE parameter to "yes".

You can specify many more parameters in the parameter file that you use with the MQParms command than you can in the response file that you use directly with the msiexec command. Also, as well as parameters that the IBM WebSphere MQ installation uses, you can specify parameters that can be used by the Prepare IBM WebSphere MQ wizard.

If you do not complete the Prepare WebSphere MQ Wizard directly after IBM WebSphere MQ installations or if for any reason your machine is rebooted between completing IBM WebSphere MQ installation and completing the Prepare WebSphere MQ Wizard, ensure that the wizard is run with Administrator privilege afterward, otherwise the installation is incomplete, and might fail. You might also see Open File - Security Warning dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the wizard to continue

An example of the file MQParms.ini is supplied with IBM WebSphere MQ. This file contains default installation parameters.

There are two ways to create a parameter file for installation:

- Copy and edit the file MQParms.ini that is supplied with the product, using an ASCII file editor.
- Create your own parameter file using an ASCII file editor.

About this task

To invoke installation using the MQParms command:

Procedure

1. From a command line, change to the root folder of the IBM WebSphere MQ client CD (that is, the location of the file MQParms.exe).
2. Enter the following command:

   ```
   MQParms [parameter_file] [parameters]
   ```

   where:

   parameter_file
   : is the file that contains the required parameter values. If this file is not in the same folder as MQParms.exe, specify the full path and file name. If you do not specify a parameter file, the default is MQParms.ini. For further details, see “MQParms parameter file” on page 274.

   parameters
   : are one or more command-line parameters, for a list of these, see the MSDN Command-Line Options web page.

Example

A typical example of an MQParms command is:

```
MQParms "c:\MyParamsFile.ini" /l*v c:\install.log
```

If you specify a parameter both on the command line and in the parameter file, the setting on the command line takes precedence.

If you do not specify /i, /x, /a, or /j, MQParms defaults to standard installation using the IBM WebSphere MQ Windows Installer package, IBM IBM WebSphere MQ.msi. That is, it generates the following part of the command line:

```
/i "current_folder\MSI\IBM WebSphere MQ.msi"
```

MQParms parameter file:

A parameter file is an ASCII text file that contains sections (stanzas) with parameters that can be used by the MQParms command. Typically, this is an initialization file such as MQParms.ini.

The MQParms command takes parameters from the following stanzas in the file:

[MSI] Contains general properties related to how the MQParms command runs and to the installation of IBM WebSphere MQ.

The properties that you can set in this stanza are listed in “Advanced installation using msiexec” on page 266, and Table 49 on page 275.

MQParms ignores any other stanzas in the file.

The stanza parameters are in the form property=value, where property is always interpreted as uppercase, but value is case sensitive. If a value string includes a blank, it must be enclosed in double quotation marks. Most other values can be enclosed in double quotation marks. Some properties can take more than one value, for example:

```
ADDLOCAL="Server,Client"
```

To clear a property, set its value to an empty string, for example:

```
REINSTALL=""
```

The following tables show the properties that you can set. The default is shown in bold.

For the [MSI] stanza, you can enter standard MSI command line options and properties. For example:

```
- /q
- ADDLOCAL="client"
- REBOOT=Suppress
```

Refer to Table 49 on page 275, and Table 50 on page 275 for the properties used to install IBM WebSphere MQ.

Table 49 on page 275 shows additional properties in the stanza that affect how the MQParms command runs, but that do not affect the installation.

Table 49. Properties used by MQParms in the MSI stanza

Property	Values	Description
MQPLOG	*path*|*file_name*	MQParms generates a text log file with the specified name and location.
MQPLANGUAGE	**system**|user| *transform_value*|existing	The installation language. system. Install using the language of the default system locale (the default). user. Install using the language of the default locale of the user. *transform_value*. Install using the language specified by this value. See Table 50. existing. If MQ already exists on the system, the same language will be used by default, otherwise system is used.
MQPSMS	**0**|no	0 or no. MQParms does not wait for the `msiexec` command to end (the default). Any other value. MQParms waits for the `msiexec` command to end.
MQPINUSE	**0**|1	If MQPINUSE is set to 1, MQParams continues installing even if IBM WebSphere MQ files are in use. If this option is used a reboot will be required to complete the installation.

Table 50. Valid values for the MQPLANGUAGE property

Language	Valid values		
U.S. English	English	en_us	1033
German	German	de_de	1031
French	French	fr_fr	1036
Spanish	Spanish	es_es	1034
Italian	Italian	it_it	1040
Brazilian Portuguese		pt_br	1046
Japanese	Japanese	ja_jp	1041
Korean	Korean	ko_kr	1042
Simplified Chinese		zh_cn	2052
Traditional Chinese		zh_tw	1028
Czech	Czech	cs_cz	1029
Russian	Russian	ru_ru	1049
Hungarian	Hungarian	hu_hu	1038
Polish	Polish	pl_pl	1045

A typical example of a parameter file is:

```
[MSI]
MQPLANGUAGE=1033
MQPLOG=%temp%\MQParms.log
MQPSMS=no
```

```
ADDLOCAL=CLIENT
/m miffile
REMOVE=""
/l*v c:\install.log
```

Modifying the client installation on Windows:

You modify the installation when WebSphere MQ for Windows client is installed and you want to remove or install some WebSphere MQ client features.

1. Insert WebSphere MQ client DVD into the DVD drive.
2. If autorun is installed, the installation process starts.

   Otherwise, double-click **Setup** in the root folder of the DVD to start the installation process.

   The WebSphere MQ client Setup window is displayed. Click **Next** to continue.
3. Select **Modify**, then click **Next**.

   The Features panel is displayed.
4. To change the installation of a feature:
   a. Click the symbol next to the feature name to display a menu.
   b. Select the required option from:
      - Install this feature
      - Install this feature and all its subfeatures (if any)
      - Do not install this feature (remove if already installed).

      The symbol next to the feature name changes to show the current installation option.
5. When your selections are complete, click **Next**.
6. The WebSphere MQ client Setup window displays a summary of the installation you selected.

   To continue, click **Modify**.
7. Wait until the progress bar is complete.

   When the WebSphere MQ client is successfully installed, the WebSphere MQ client Setup window displays the following message: `Installation Wizard Completed Successfully`

   Click **Finish** to close the window.

Modifying the client installation on Windows using Add/Remove Programs:

On some versions of Windows, you can modify an installation by using Add/Remove Programs.

For Windows Active Directory Server, Windows XP, or Windows 2003, follow these steps. You cannot use this method to modify an installation on any other version of Windows.

1. From the Windows taskbar, select **Start** > **Settings** > **Control Panel**.
2. Select **Add/Remove Programs**.
3. Select **IBM WebSphere MQ**.
4. Select **Change**.

   The IBM WebSphere MQ Setup window with the Program Maintenance panel is displayed. Follow the procedure for modifying the installation by using the process from step 3 to the end.

Silently modifying a WebSphere MQ client installation using msiexec:

To silently modify an installation using msiexec, follow the instructions on the installation pages, but set the ADDLOCAL parameter to include the features you want to add, and set the REMOVE parameter to the features you want to remove.

For example if you used ADDLOCAL="JavaMsg" and REMOVE="" it would modify the installation to include the Java Messaging and Web Services feature.

The instructions for msiexec begin here: “Advanced installation using msiexec” on page 266

Silently modifying a WebSphere MQ client installation using MQParms:

To silently modify an installation using MQParms, follow the instructions on the installation pages, but set the ADDLOCAL parameter to include the features you want to add, and set the REMOVE parameter to the features you want to remove.

For example if you used ADDLOCAL="JavaMsg" and REMOVE="" it would modify the installation to include the Java Messaging and Web Services feature.

For details of the MQParms command, see “Using the MQParms command” on page 239.

Installing WebSphere MQ Advanced Message Security

Install and uninstall IBM WebSphere MQ Advanced Message Security component.

Before you begin

WebSphere MQ Advanced Message Security is a separately installed and licensed component of WebSphere MQ and is another option on the WebSphere MQ installer. Make sure that you purchase a license for using WebSphere MQ Advanced Message Security before the installation.

Additionally, make sure the following WebSphere MQ components are installed in your environment:

- MQSeriesRuntime
- MQSeriesServer

Related tasks:

“Installing WebSphere MQ Advanced Message Security on AIX”
You can install IBM WebSphere MQ Advanced Message Security component on AIX platforms using either system management interface tool (SMIT) or the command line.

“Installing WebSphere MQ Advanced Message Security on HP-UX” on page 278
You can install IBM WebSphere MQ Advanced Message Security component on HP-UX platforms.

“Installing WebSphere MQ Advanced Message Security on Linux” on page 278
You can install IBM WebSphere MQ Advanced Message Security on Linux platforms.

“Installing WebSphere MQ Advanced Message Security on Windows” on page 279
Once you purchase the IBM WebSphere MQ Advanced Message Security license, you can install the component on Windows platforms.

“Uninstalling WebSphere MQ Advanced Message Security” on page 315
Information provided guides you through the uninstallation process of IBM WebSphere MQ Advanced Message Security component.

Installing WebSphere MQ Advanced Message Security on AIX

You can install IBM WebSphere MQ Advanced Message Security component on AIX platforms using either system management interface tool (SMIT) or the command line.

Installing using SMIT:
Procedure

1. Log on as root.
2. Change the directory to the location of the installation packages.
3. Start the system management interface tool (SMIT). The system management menu is displayed.
4. Select the required SMIT window using the following sequence:

```
Software Installation and Maintenance
        Install and Update Software
   Install Software
```

5. Enter the directory location of the installation package.
6. Press F4 to list the software in the **SOFTWARE name** option.
7. Select the `mqm.ams.rte` and press Enter.
8. Accept the default setting for the remaining options and press Enter.

Results

WebSphere MQ Advanced Message Security has been installed successfully.

Installing using command line:

Procedure

1. Log on as root.
2. Set your current directory to the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. Run the following command:
   ```
   installp -a -c -Y -d. mqm.ams.rte
   ```
   Note the period, signifying the current directory, following the `-d` parameter.

Results

WebSphere MQ Advanced Message Security component has been installed successfully.

Installing WebSphere MQ Advanced Message Security on HP-UX

You can install IBM WebSphere MQ Advanced Message Security component on HP-UX platforms.

Procedure

1. Log on as root.
2. Set your current directory to the location of the installation file. The location might be the mount point of the DVD, a network location, or a local file system directory.
3. In the command line, issue the following command:
   ```
   swinstall -s MQSERIES.MQM-AMS
   ```

Results

WebSphere MQ Advanced Message Security component has been installed successfully.

Installing WebSphere MQ Advanced Message Security on Linux

You can install IBM WebSphere MQ Advanced Message Security on Linux platforms.

Procedure

1. Log on as root.
2. Set your current directory to the location of the installation file. The location might be the mount point of the server CD, a network share, or a local file system directory.
3. If this installation is not the first installation on the system, you must run the **`crtmqpkg`** command to create a unique set of packages to install on the system. In order for the **`crtmqpkg`** command to run on Linux, the **`pax`** command or **`rpmbuild`** must be installed.

   Important: **`pax`** and **`rpmbuild`** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

   a. Enter the following command:
      ```
      ./crtmqpkg suffix
      ```

where *suffix* is a name of your choosing, that uniquely identifies the installation packages on the system. *suffix* is not the same as an installation name, although the names can be identical. *suffix* is limited to 16 characters in the ranges A-Z, a-z, and 0-9.

Note: This command creates a full copy of the installation packages in a subdirectory of `/var/tmp`. You must ensure that the system has enough space before running the command.

b. Set your current directory to the location specified when the **`crtmqpkg`** command completes. This directory is a subdirectory of `/var/tmp/mq_rpms`, in which the unique set of packages is created. The packages have the *suffix* value contained within the filename. For example, using a suffix of "1":

```
./crtmqpkg 1
```

there will be a subdirectory named `/var/tmp/mq_rpms/1/i386` and the packages will be renamed, for example:

```
From: MQSeriesAMS-7.5.0-0.i386.rpm
To: MQSeriesAMS_1-7.5.0-0.i386.rpm
```

4. In the command line, issue the following command:

   This example shows a minimum installation:

   ```
   rpm -iv <package_name>
   ```

   where `<package_name>` is one of the following:

   - MQSeriesAMS-7.5.0-0.i386.rpm
   - MQSeriesAMS-7.5.0-0.x86_64.rpm
   - MQSeriesAMS-7.5.0-0.ppc.rpm
   - MQSeriesAMS-7.5.0-0.s390.rpm

Results

WebSphere MQ Advanced Message Security has been installed successfully.

Installing WebSphere MQ Advanced Message Security on Windows

Once you purchase the IBM WebSphere MQ Advanced Message Security license, you can install the component on Windows platforms.

Using the Launchpad:
Procedure

1. Access the WebSphere MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `setup.exe` in the base directory of the WebSphere MQ installation image.
   - From a DVD, this location might be:

     `*E:*\setup.exe`
   - From a network location, this location might be:

     `*m:\instmqs*\setup.exe`
   - From a local file system directory, this location might be:

     `*C:\instmqs*\setup.exe`
3. Double-click the **Setup** icon to start the installation process. It is possible to start the process by either:
   - Running `setup.exe` from the command prompt.
   - Double-clicking `setup.exe` from WebSphere MQ Explorer.

Note: If you are installing on a Windows system with UAC enabled, accept the Windows prompt to allow the launchpad to run as elevated. During installation, you might also see Open File - Security Warning dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the installation to continue.

The WebSphere MQ Installation Launchpad window is displayed.

4. Continue to follow the Launchpad instructions as shown on screen.

Verifying a WebSphere MQ installation

The topics in this section provide instructions on how to verify a server or a client installation of IBM WebSphere MQ on Windows, UNIX and Linux, and HP Integrity NonStop Server systems.

To verify a server installation, either using the command line or using the postcard application, see "Verifying a server installation."

To verify a client installation, either using the command line or using the IBM WebSphere MQ Explorer, see "Verifying a client installation" on page 290.

Related concepts:

"Installing IBM WebSphere MQ" on page 209
The topics in this section provide instructions on how to install IBM WebSphere MQ.

"Uninstalling" on page 304
The topics in this section provide instructions on how to uninstall components.

Verifying a server installation

You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

You can use either the command line or the postcard application to verify your installation. The postcard application is Java based and requires a system with the ability to view a graphical display.

A local installation uses a single queue manager while a server-to-server installation has multiple queue managers and queues, and both sender and receiver channels.

For a server-to-server verification, the communication links between the two systems must be checked. Before you can do the verification, you must ensure that the communications protocol is installed and configured on both systems. The examples explain how to verify your installation using TCP.

UNIX systems
IBM WebSphere MQ supports both TCP and SNA. If you do not use TCP, see Setting up communication on UNIX and Linux systems.

Linux IBM WebSphere MQ for Linux supports TCP on all Linux platforms. On x86 platforms and Power platforms, SNA is also supported. If you want to use the SNA LU6.2 support on these platforms, you need the IBM Communications Server for Linux Version 6.2. The Communications Server is available as a PRPQ product from IBM. For more details, see http://www.ibm.com/software/network/commserver/about.

If you do not use TCP, see Setting up communication on UNIX and Linux systems.

Windows
IBM WebSphere MQ for Windows supports TCP, SNA, NetBios, and SPX. If you do not use TCP, see Setting up communication for Windows.

Related concepts:

"Verify the installation using the command line"
You can use the command line to verify a local installation or a server-to-server installation.

"Verify the installation using the Postcard application" on page 286
You can set up and use the Postcard application to verify a local installation or a server-to-server installation.

Related tasks:

"Verifying a local installation using the command line"
You can verify a local installation by using the command line to create a simple configuration of one queue manager and one queue.

"Verifying a server-to-server installation using the command line" on page 283
You can verify a server-to-server installation using two servers, one as a sender and one as a receiver.

"Using the Postcard application to verify a local installation" on page 287
Sending messages successfully between two Postcard applications verifies a local installation.

"Using the Postcard application to verify a server-to-server installation" on page 288
You can use two instances of the Postcard application to verify that a server-to-server installation is working.

Verify the installation using the command line

You can use the command line to verify a local installation or a server-to-server installation.

Use the command line to verify that IBM WebSphere MQ is successfully installed, and that the associated communication links are working properly.

You can also verify an installation using the postcard application. The postcard application is Java based and requires a system with the ability to view a graphical display. See "Verify the installation using the Postcard application" on page 286.

Related tasks:

"Verifying a local installation using the command line"
You can verify a local installation by using the command line to create a simple configuration of one queue manager and one queue.

"Verifying a server-to-server installation using the command line" on page 283
You can verify a server-to-server installation using two servers, one as a sender and one as a receiver.

Verifying a local installation using the command line:

You can verify a local installation by using the command line to create a simple configuration of one queue manager and one queue.

Before you begin

To verify the installation, you must first install the samples package.

Before beginning the verification procedure, you might want to check that you have the latest fixes for your system. For more information about where to find the latest updates, see "Finding the latest information" on page 190.

About this task

Use the following steps to configure your default queue manager from the command line. After the queue manager is configured, use the **amqsput** sample program to put a message on the queue. You then use the **amqsget** sample program to get the message back from the queue.

This procedure is for Windows, UNIX and Linux systems.

IBM WebSphere MQ object definitions are case-sensitive. Any text entered as an MQSC command in lowercase is converted automatically to uppercase unless you enclose it in single quotation marks. Make sure that you type the examples exactly as shown.

Procedure

1. If you are verifying an installation on a UNIX or Linux system, log in as a user in the `mqm` group.
2. Set up your environment:
   a. Set up environment variables for use with a particular installation by entering one of the following commands:
      - On Windows:
        ```
        MQ_INSTALLATION_PATH/bin/setmqenv -s
        ```
        where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
      - On UNIX systems:
        ```
        . MQ_INSTALLATION_PATH/bin/setmqenv -s
        ```
        where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
      - On Linux systems:
        ```
        . source/ MQ_INSTALLATION_PATH/bin/setmqenv -s
        ```
        where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

   b. Check that the environment is set up correctly by entering the following command:
      ```
      dspmqver
      ```
      If the command completes successfully, and the expected version number and installation name are returned, the environment is set up correctly.
3. Create a queue manager called `QMA`by entering the following command:
   ```
   crtmqm QMA
   ```
   Messages indicate when the queue manager is created, and when the default IBM WebSphere MQ objects are created.
4. Start the queue manager by entering the following command:
   ```
   strmqm QMA
   ```
   A message indicates when the queue manager starts.
5. Start MQSC by entering the following command:
   ```
   runmqsc QMA
   ```
   A message indicates when MQSC starts. MQSC has no command prompt.
6. Define a local queue called `QUEUE1` by entering the following command:
   ```
   DEFINE QLOCAL (QUEUE1)
   ```
   A message indicates when the queue is created.
7. Stop MQSC by entering the following command:
   ```
   end
   ```
   Messages are shown, followed by the command prompt.

Note: Subsequent steps require that the samples package is installed.

8. If you are verifying an installation on a UNIX or Linux system, change into the *MQ_INSTALLATION_PATH*/samp/bin directory, which contains the sample programs. *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed.
9. Put a message on the queue by entering one of the following commands:
   - On UNIX and Linux systems:
     ```
     ./amqsput QUEUE1 QMA
     ```
   - On Windows systems:
     ```
     amqsput QUEUE1 QMA
     ```

   The following messages are shown:
   ```
   Sample AMQSPUT0 start
   target queue is QUEUE1
   ```
10. Type some message text on one or more lines, where each line is a different message. Enter a blank line to end the message input. The following message is shown:
    ```
    Sample AMQSPUT0 end
    ```

    Your messages are now on the queue and the command prompt is shown.
11. Get the messages from the queue, by entering one of the following commands:
    - On UNIX and Linux systems:
      ```
      ./amqsget QUEUE1 QMA
      ```
    - On Windows systems:
      ```
      amqsget QUEUE1 QMA
      ```

    The sample program starts, and your messages are displayed.

Results

You have successfully verified your local installation.

Verifying a server-to-server installation using the command line:

You can verify a server-to-server installation using two servers, one as a sender and one as a receiver.

Before you begin

- Make sure that TCP/IP and WebSphere MQ are installed on both servers.
- Make sure that you are a member of the WebSphere MQ administrators group (**mqm**) on each server.
- Decide which installation is the sender server and which installation is the receiver server. The installations might be on the same system, or on different systems.

About this task

This procedure provides instructions for Windows, UNIX and Linux systems only.

IBM WebSphere MQ object definitions are case-sensitive. Any text entered as an MQSC command in lowercase is converted automatically to uppercase unless you enclose it in single quotation marks. Make sure that you type the examples exactly as shown.

Procedure

1. On the **receiver** server:
   a. If the receiver server is a UNIX or Linux system, log in as a user in the mqm group.
   b. Check which ports are free, for example by running **netstat**. For more information about this command, see the documentation of your operating system.

If port 1414 is not in use, make a note of 1414 to use as the port number in step 2 h. Use the same number for the port for your listener later in the verification. If it is in use, note a port that is not in use; for example 1415.

c. Set up the environment for the installation you are using by entering one of the following commands at the command prompt:
   - On Windows:

     ```
     MQ_INSTALLATION_PATH\bin\setmqenv -s
     ```

     where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
   - On UNIX and Linux systems:

     ```
     . MQ_INSTALLATION_PATH/bin/setmqenv -s
     ```

     where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

d. Create a queue manager called `QMB` by entering the following command at the command prompt:

   ```
   crtmqm QMB
   ```

   Messages tell you that the queue manager has been created, and that the default IBM WebSphere MQ objects have been created.

e. Start the queue manager by entering the following command:

   ```
   strmqm QMB
   ```

   A message tells you when the queue manager has started.

f. Start MQSC by entering the following command:

   ```
   runmqsc QMB
   ```

   A message tells you that MQSC has started. MQSC has no command prompt.

g. Define a local queue called `RECEIVER.Q` by entering the following command:

   ```
   DEFINE QLOCAL (RECEIVER.Q)
   ```

   A message tells you the queue has been created.

h. Define a listener by entering the following command:

   ```
   DEFINE LISTENER (LISTENER1) TRPTYPE (TCP) CONTROL (QMGR) PORT (PORT_NUMBER)
   ```

   Where *port_number* is the name of the port the listener runs on. This number must be the same as the number used when defining your sender channel.

i. Start the listener by entering the following command:

   ```
   START LISTENER (LISTENER1)
   ```

   Note: Do not start the listener in the background from any shell that automatically lowers the priority of background processes.

j. Define a receiver channel by entering the following command:

   ```
   DEFINE CHANNEL (QMA.QMB) CHLTYPE (RCVR) TRPTYPE (TCP)
   ```

   A message tells you when the channel has been created.

k. End MQSC by typing:

   ```
   end
   ```

   Some messages are displayed, followed by the command prompt.

2. On the **`sender`** server:

   a. If the sender server is a UNIX or Linux system, log in as a user in the `mqm` group.

b. Set up the environment for the installation you are using by entering one of the following commands at the command prompt:
   - On Windows:

     ```
     MQ_INSTALLATION_PATH\bin\setmqenv -s
     ```

     where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
   - On UNIX and Linux systems:

     ```
     . MQ_INSTALLATION_PATH/bin/setmqenv -s
     ```

     where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

c. Create a queue manager called `QMA` by entering the following command at the command prompt:

   ```
   crtmqm QMA
   ```

   Messages tell you that the queue manager has been created, and that the default IBM WebSphere MQ objects have been created.

d. Start the queue manager, by entering the following command:

   ```
   strmqm QMA
   ```

   A message tells you when the queue manager has started.

e. Start MQSC by entering the following command:

   ```
    runmqsc QMA
   ```

   A message tells you that an MQSC session has started. MQSC had no command prompt.

f. Define a local queue called `QMB` (to be used as a transmission queue) by entering the following command:

   ```
   DEFINE QLOCAL (QMB) USAGE (XMITQ)
   ```

   A message tells you when the queue has been created.

g. Define a local definition of the remote queue with by entering the following command:

   ```
   DEFINE QREMOTE (LOCAL.DEF.OF.REMOTE.QUEUE) RNAME (RECEIVER.Q)
   RQMNAME ('QMB') XMITQ (QMB)
   ```

h. Define a sender channel by entering one of the following commands:

   con-name is the TCP/IP address of the receiver system. If both installations are on the same system, the *con-name* is `localhost`. *port* is the port you noted in 1 b. If you do not specify a port, the default value of 1414 is used.

   ```
   DEFINE CHANNEL (QMA.QMB) CHLTYPE (SDR)
   CONNAME ('CON-NAME(PORT)') XMITQ (QMB) TRPTYPE (TCP)
   ```

i. Start the sender channel by entering the following command:

   ```
   START CHANNEL(QMA.QMB)
   ```

   The receiver channel on the receiver server starts automatically when the sender channel starts.

j. Stop MQSC by entering the following command:

   ```
   end
   ```

   Some messages are displayed, followed by the command prompt.

k. If the sender server is a UNIX or Linux system, change into the `MQ_INSTALLATION_PATH/samp/bin` directory. This directory contains the sample programs. *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed.

l. If both the sender server and receiver server are installations on the same system, check that the queue managers have been created on different installations by entering the following command:

   ```
   dspmq -o installation
   ```

If the queue managers are on the same installation, move either QMA to the sender installation or QMB to the receiver installation by using the **setmqm** command. For more information, see setmqm.

m. Put a message on the local definition of the remote queue, which in turn specifies the name of the remote queue. Enter one of the following commands:
   - On Windows:
     ```
     amqsput LOCAL.DEF.OF.REMOTE.QUEUE QMA
     ```
   - On UNIX and Linux systems:
     ```
     ./amqsput LOCAL.DEF.OF.REMOTE.QUEUE QMA
     ```

   A message tells you that `amqsput` has started.

n. Type some message text on one or more lines, followed by a blank line. A message tells you that `amqsput` has ended. Your message is now on the queue and the command prompt is displayed again.

3. On the **receiver** server:

   a. If your receiver server is a UNIX or Linux system, change into the *MQ_INSTALLATION_PATH*/samp/bin directory. This directory contains the sample programs. *MQ_INSTALLATION_PATH* represents the high-level directory in which IBM WebSphere MQ is installed.

   b. Get the message from the queue on the receiver by entering one of the following commands:
      - On Windows:
        ```
        amqsget RECEIVER.Q QMB
        ```
      - On UNIX and Linux systems:
        ```
        ./amqsget RECEIVER.Q QMB
        ```

      The sample program starts, and your message is displayed. After a pause, the sample ends. Then the command prompt is displayed.

Results

You have now successfully verified the server-to-server installation.

Verify the installation using the Postcard application

You can set up and use the Postcard application to verify a local installation or a server-to-server installation.

Use the Postcard application to verify that IBM WebSphere MQ is successfully installed, and that the associated communication links are working properly.

The postcard application is Java based and requires a system with the ability to view a graphical display. You can also verify an installation using the command line, see "Verify the installation using the command line" on page 281.

Note: Using Postcard to verify a WebSphere MQ installation is only possible if there is one IBM WebSphere MQ installation on that box. The Default Configuration wizard will not create a default configuration if a queue manager already exists on the box. The Default Configuration wizard will run on any installation on a box but only one default configuration can be created per box. Using Postcard to verify second and subsequent installations of IBM WebSphere MQ on the same box is not possible.

Related tasks:

"Using the Postcard application to verify a local installation"
Sending messages successfully between two Postcard applications verifies a local installation.

"Using the Postcard application to verify a server-to-server installation" on page 288
You can use two instances of the Postcard application to verify that a server-to-server installation is working.

Using the Postcard application to verify a local installation:

Sending messages successfully between two Postcard applications verifies a local installation.

Before you begin

To verify that the local installation is working, you can run two instances of the Postcard application on the same server. The postcard application can send messages to, and receive messages from, other postcard applications. Successful sending and receiving of messages verifies that IBM WebSphere MQ is installed and working correctly on the server.

Note:

- If the system has multiple IBM WebSphere MQ installations, ensure that Postcard has not been run before on any installations on that server. As the default configuration can only exist on one IBM WebSphere MQ installation per system, the Default Configuration wizard and Postcard can not be used for verification of a second or any subsequent installation.
- The Postcard application has a graphical interface. To view this interface, your system requires the ability to view a graphical display.
- Before you can run the Postcard application, you must ensure that you are a member of the IBM WebSphere MQ administrators group (**mqm**).

Procedure

1. If you are verifying an installation on UNIX, Linux or Windows systems, log on as a user in group **mqm**.
2. Start the postcard application in one of the following ways:
   a. From the command line:
      1) Change the directory to *MQ_INSTALLATION_PATH*/java/bin. *MQ_INSTALLATION_PATH* represents the high-level directory in which IBM WebSphere MQ is installed.
      2) Run the postcard application by entering one of the following commands:
         - UNIX and Linux systems:
           ```
           ./postcard
           ```
         - Windows systems:
           ```
           postcard
           ```
   b. From the IBM WebSphere MQ Explorer: On Windows and Linux systems (x86-64 platforms), you can start MQ Explorer by using the system menu, the **MQExplorer** command (preferred command), or the MQExplorer executable file. The **strmqcfg** command is still usable.
      1) If the Welcome to IBM WebSphere MQ Explorer Content view page does not show, click **IBM WebSphere MQ** in the Navigator view to show the Welcome page.
      2) Click **Launch Postcard** to start the Postcard.
3. At the Postcard - Sign On window, type in a nickname to use to send messages within the Postcard application (for example, User1).
4. Select the queue manager to use as the mailbox:

- If you do not have any queue managers, you are prompted to either launch the Default Configuration or close the Postcard application. Launching the Default Configuration creates a default queue manager.
- If the only queue manager on your server is the default queue manager, this queue manager is used automatically for the postcard application. The default queue manager is created by running the Default Configuration wizard
- If you have created your own queue managers, but you have not run the Default Configuration wizard, select an appropriate queue manager from the list.
- If you have run the Default Configuration wizard and you want to use the default queue manager, but there are other queue managers on your server, select the **Advanced** check box. Then select **Use Default Configuration as mailbox**.
- If you have run the Default Configuration wizard and also created your own queue managers, and you do not want to use the default queue manager, select the **Advanced** check box. Then select **Choose queue manager as mailbox**, and then select the appropriate queue manager from the list.

When your selection is complete, click **OK** to display your first Postcard window.

5. Run a second instance of the Postcard application by following the steps used to open the first instance of the Postcard application.
6. The Postcard - Sign On panel is displayed again. Type in a second nickname to use to send messages within this second Postcard application (for example, `User2`).
7. Repeat the selection of the queue manager that you want to use as the mailbox (as described in step 4). The queue manager you select for this second Postcard must be the same queue manager as used for the first instance of the Postcard application.
8. In the first Postcard, (User1), enter the nickname (`User2`) for the second Postcard application in the **To:** field. Because the sender and receiver are on the same server, you can leave the **On:** field blank.
9. Type a message in the **Message:** field and click **Send**.
10. The **Postcards sent and received** area of the Postcard shows details of the message. In the sending Postcard, the message is displayed as sent. In the receiving Postcard, the message is displayed as received.
11. In the receiving Postcard, (User2), double-click the message in the **Postcards sent and received** area to view it. When this message arrives, it verifies that IBM WebSphere MQ is correctly installed.

What to do next

Depending on your situation, you might want to do the following tasks:

- Install IBM WebSphere MQ on other servers. Follow the installation procedure for the appropriate platform. Ensure that you use the **Join Default Cluster** window in the Default Configuration wizard to add the other servers to the cluster on your first server.
- Install the IBM WebSphere MQ MQI client on other servers. See “Installing a IBM WebSphere MQ client” on page 256.
- Continue with further administration tasks, see Administering IBM WebSphere MQ.

Using the Postcard application to verify a server-to-server installation:

You can use two instances of the Postcard application to verify that a server-to-server installation is working.

Before you begin

You can use the Postcard application on two servers, one instance of the Postcard application on each server, to verify that a server-to-server installation is working. Successful sending and receiving of messages verifies that IBM WebSphere MQ is successfully installed, and that communication between the two servers is working correctly.

Note:

- If the system has multiple WebSphere MQ installations, ensure that Postcard has not been run before on any installations on that server. As the default configuration can only exist on one WebSphere MQ installation per system, the Default Configuration wizard and Postcard can not be used for verification of a second or any subsequent installation.
- The two server installations must be on different systems to do a server-to-server verification using the postcard application. To verify a server-to-server installation on the same machine, you can use the command line. See "Verifying a server-to-server installation using the command line" on page 283
- Make sure that TCP/IP and IBM WebSphere MQ are installed on both servers.
- Make sure that your systems are able to view a graphical display.
- Make sure that you are a member of the IBM WebSphere MQ administrators group (**mqm**) on each server.
- Check that one of the following scenarios applies:
  - Neither server has had any queue managers created.
  - Use the Default Configuration wizard to create default queue managers on each server and link them to the default cluster. Details on how to use the Default Configuration wizard are provided in this topic.
  - Both servers have existing queue managers and these queue managers are in the same cluster.

    If your queue managers are not in the same cluster, create new queue managers on both servers. Then create a cluster, and ensure that the queue managers that you create on each server belong to that cluster.
  - You have configured channels to communicate between the two servers.

    For instructions on how to set up the channels, see "Verifying a server-to-server installation using the command line" on page 283. After you have set up the channels, follow the instructions in this topic to verify your server-to-server installation.

Procedure

1. If you are verifying an installation on UNIX or Linux systems, log on as a user in group **mqm**.
2. Start the postcard application in one of the following ways:
   a. From the command line:
      1) Change the directory to *MQ_INSTALLATION_PATH*/java/bin. *MQ_INSTALLATION_PATH* represents the high-level directory in which IBM WebSphere MQ is installed.
      2) Run the postcard application by entering one of the following commands:
         - UNIX and Linux systems:

           ```
           ./postcard
           ```
         - Windows systems:

           ```
           postcard
           ```
   b. From the IBM WebSphere MQ Explorer: On Windows and Linux systems (x86-64 platforms), you can start MQ Explorer by using the system menu, the `MQExplorer` executable file, or the **strmqcfg** command.
      1) If the Welcome to IBM WebSphere MQ Explorer Content view page does not show, click **IBM WebSphere MQ** in the Navigator view to show the Welcome page.
      2) Click **Launch Postcard** to start the Postcard.

3. At the Postcard - Sign On window, type a nickname to use to send messages within the Postcard application. For example, `User1` for the first server, and `User2` for the second server.
4. When you have completed the wizard, you are taken back to the Postcard - Sign On window.
5. Select the queue manager to use as the mailbox:
   - If you do not have any queue managers, you are prompted to either launch the Default Configuration or close the Postcard application. Work through the Default Configuration wizard. When you get to the option to join the queue manager to the default cluster, tick the check box. On the next screen:
     - For the first server, select **yes, make it the repository for the cluster**.
     - For the second server, select **No another computer has already joined the cluster as a repository**. When requested, enter the location of the repository, by typing the name of the sender server.
   - If the only queue manager on your server is the default queue manager, this queue manager is used automatically for the postcard application. The default queue manager is created by running the Default Configuration wizard
   - If you have created your own queue managers, but you have not run the Default Configuration wizard, select an appropriate queue manager from the list.
   - If you have run the Default Configuration wizard and you want to use the default queue manager, but there are other queue managers on your server, select the **Advanced** check box. Then select **Use Default Configuration as mailbox**.
   - If you have run the Default Configuration wizard and also created your own queue managers, and you do not want to use the default queue manager, select the **Advanced** check box. Then select **Choose queue manager as mailbox**, and then select the appropriate queue manager from the list.

   When your selection is complete, click **OK**.
6. Complete steps 1 - 5 for the second server.
7. In the Postcard on the first server:
   a. Enter the nickname (`user2`) for the Postcard application on the second server in the **To:** field.
   b. Enter the queue manager on the second server in the **On:** field.
   c. Type a message in the **Message:** field and click **Send**.
8. In the Postcard on the second server:
   a. In the **Postcards sent and received**, double-click the message marked as received to view the message from the first server.
   b. Optional: Send a postcard to the first server by adapting the instructions in step 7. You must enter details of the first server in the **To:** field and the **On:** field.

   The messages verify that IBM WebSphere MQ is correctly installed and that your communication link between the two servers is working correctly.

Verifying a client installation

You can verify that your WebSphere MQ MQI client installation completed successfully and that the communication link is working.

The verification procedure shows how to create a queue manager called `queue.manager.1`, a local queue called `QUEUE1`, and a server-connection channel called `CHANNEL1` on the server.

It shows how to create the client-connection channel on the WebSphere MQ MQI client workstation. It then shows how to use the sample programs to put a message onto a queue, and get the message from the queue.

The example does not address any client security issues. See Setting up WebSphere MQ MQI client security for details if you are concerned with WebSphere MQ MQI client security issues.

The verification procedure assumes that:

- The full WebSphere MQ server product has been installed on a server.
- The server installation is accessible on your network.
- The WebSphere MQ MQI client software has been installed on a client system.
- The WebSphere MQ sample programs have been installed.
- TCP/IP has been configured on the server and client systems. For more information, see Configuring connections between the server and client.

To begin the verification procedure using the command line, see "Verifying a client installation using the command line."

To begin the verification procedure for the Windows and Linux operating systems when you are using the IBM WebSphere MQ Explorer, see "Verifying a client installation using WebSphere MQ Explorer" on page 296.

Related tasks:

"Setting up the server using the command line" on page 292
Follow these instructions to create a queue manager, queue, and channel on the server. You can then use these objects to verify the installation.

"Connecting a WebSphere MQ MQI client to a queue manager, using the MQSERVER environment variable" on page 294
When a IBM WebSphere MQ application is run on the IBM WebSphere MQ MQI client, it requires the name of the MQI channel, the communication type, and the address of the server to be used. Provide these parameters by defining the MQSERVER environment variable.

"Setting up the server using WebSphere MQ Explorer" on page 296
You can use the WebSphere MQ Explorer to create a queue manager, queue and server-connection channel on Windows and Linux systems .

"Setting up the client using WebSphere MQ Explorer" on page 297
You can use IBM WebSphere MQ Explorer to define the client-connection if you are setting up the client and server on the same workstation on a Windows or Linux system.

"Testing communication between a client and a server" on page 298
On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

Verifying a client installation using the command line

You can verify a client installation using the command line. On the server you create a queue manager, a local queue, a listener, and a server-connection channel. You must also apply security rules to allow the client to connect and make use of the queue defined. On the client you create a client-connection channel, and then use the sample PUT and GET programs to complete the verification procedure.

First, set up the server using the command line, using the instructions in "Setting up the server using the command line" on page 292.

Once you have set up the server, you must set up the client, using the instructions in "Connecting a WebSphere MQ MQI client to a queue manager, using the MQSERVER environment variable" on page 294.

Finally, you can test the communications between client and server, using the instructions in "Testing communication between a client and a server" on page 298.

Related concepts:

"Verifying a client installation using WebSphere MQ Explorer" on page 296
You can verify a client installation using the WebSphere MQ Explorer on Windows and Linux. On the server, you create a queue manager, a local queue, a listener and a server-connection channel. On the client system you create a client-connection channel. Then from the command line you use the sample PUT and GET programs to complete the verification procedure.

"Installing a IBM WebSphere MQ client" on page 256

"Verifying a server installation" on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Setting up the server using the command line:

Follow these instructions to create a queue manager, queue, and channel on the server. You can then use these objects to verify the installation.

About this task

These instructions assume that no queue manager or other IBM WebSphere MQ objects have been defined.

IBM WebSphere MQ object definitions are case-sensitive. Any text entered as an MQSC command in lowercase is converted automatically to uppercase unless you enclose it in single quotation marks. Make sure that you type the examples exactly as shown.

Procedure

1. Create a user ID on the server that is not in the `mqm` group.
2. If your server is on a UNIX or Linux system, log in as a user in the mqm group.
3. If you are verifying an installation on a Windows, UNIX or Linux system, you must set various environment variables so that the installation can be used in the current shell. You can set the environment variables by entering one of the following commands:
   - On Windows:
     ```
     . MQ_INSTALLATION_PATH\bin\setmqenv -s
     ```
     where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
   - On UNIX and Linux systems:
     ```
     . MQ_INSTALLATION_PATH/bin/setmqenv -s
     ```
     where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
4. Create a queue manager called `QUEUE.MANAGER.1` by entering the following command:
   ```
   crtmqm QUEUE.MANAGER.1
   ```
   You see messages telling you that the queue manager has been created.
5. Start the queue manager by entering the following command:
   ```
   strmqm QUEUE.MANAGER.1
   ```
   A message tells you when the queue manager has started.
6. Start MQSC by entering the following command:
   ```
   runmqsc QUEUE.MANAGER.1
   ```

A message tells you that an MQSC session has started. MQSC has no command prompt.

7. Define a local queue called QUEUE1 by entering the following command:

   ```
   DEFINE QLOCAL (QUEUE1)
   ```

   A message tells you when the queue has been created.

8. Allow the user ID that you created in step 1 to use QUEUE1 by entering the following command:

   ```
   SET AUTHREC PROFILE(QUEUE1) OBJTYPE(QUEUE) PRINCIPAL('non_mqm_user') AUTHADD(PUT,GET)
   ```

   where *non_mqm_user* is the user ID created in step 1. A message tells you when the authorisation has been set. You must also run the following command to give the user ID authority to connect:

   ```
   SET AUTHREC OBJTYPE(QMGR) PRINCIPAL('non_mqm_user') AUTHADD(CONNECT)
   ```

   If this command is not run, a 2305 stop error is returned.

9. Define a server-connection channel by entering the following command:

   ```
   DEFINE CHANNEL (CHANNEL1) CHLTYPE (SVRCONN) TRPTYPE (TCP)
   ```

   A message tells you when the channel has been created.

10. Allow your client channel to connect to the queue manager and run under the user ID that you created in step 1, by entering the following MQSC command:

    ```
    SET CHLAUTH(CHANNEL1) TYPE(ADDRESSMAP) ADDRESS('client_ipaddr') MCAUSER('non_mqm_user')
    ```

    where *client_ipaddr* is the IP address of the client system, and *non_mqm_user* is the user ID created in step 1. A message tells you when the rule has been set.

11. Define a listener by entering the following command:

    ```
    DEFINE LISTENER (LISTENER1) TRPTYPE (TCP) CONTROL (QMGR) PORT (port_number)
    ```

    where *port_number* is the number of the port the listener is to run on. This number must be the same as the number used when defining your client-connection channel in “Installing a IBM WebSphere MQ client” on page 256; see the appropriate HP-UX and Linux sections for more information.

    Note: If you omit the port parameter from the command, a default value of 1414 is used for the listener port. If you want to specify a port other than 1414, you must include the port parameter in the command, as shown.

12. Start the listener by entering the following command:

    ```
    START LISTENER (LISTENER1)
    ```

13. Stop MQSC by entering:

    ```
    end
    ```

    You see some messages, followed by the command prompt.

What to do next

Follow the instructions to set up the client. See “Connecting a WebSphere MQ MQI client to a queue manager, using the MQSERVER environment variable” on page 294.

Related concepts:

"Verifying a client installation using the command line" on page 291
You can verify a client installation using the command line. On the server you create a queue manager, a local queue, a listener, and a server-connection channel. You must also apply security rules to allow the client to connect and make use of the queue defined. On the client you create a client-connection channel, and then use the sample PUT and GET programs to complete the verification procedure.

"Verifying a client installation using WebSphere MQ Explorer" on page 296
You can verify a client installation using the WebSphere MQ Explorer on Windows and Linux. On the server, you create a queue manager, a local queue, a listener and a server-connection channel. On the client system you create a client-connection channel. Then from the command line you use the sample PUT and GET programs to complete the verification procedure.

"Installing a IBM WebSphere MQ client" on page 256

"Verifying a server installation" on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Related tasks:

"Testing communication between a client and a server" on page 298
On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

Connecting a WebSphere MQ MQI client to a queue manager, using the `MQSERVER` environment variable:

When a IBM WebSphere MQ application is run on the IBM WebSphere MQ MQI client, it requires the name of the MQI channel, the communication type, and the address of the server to be used. Provide these parameters by defining the `MQSERVER` environment variable.

Before you begin

Before you start this task, you must complete the task, "Setting up the server using the command line" on page 292, and save the following information:

- The hostname or IP address of the server and port number that you specified when creating the listener.
- The channel name of the server-connection channel.

About this task

This task describes how to connect a IBM WebSphere MQ MQI client, by defining the `MQSERVER` environment variable on the client.

If you are using Windows, HP Integrity NonStop Server, UNIX or Linux systems, you can give the client access to the generated client channel definition table, `amqclchl.tab` instead; see Accessing client-connection channel definitions .

Alternatively, on Windows, if Active Directory support is enabled, the client discovers the client-connection information dynamically from the Active Directory.

Procedure

1. Check the TCP/IP connection. From the client, enter one of the following commands:
   - `ping server-hostname`

- `ping n.n.n.n`

  `n.n.n.n` represents the network address. You can set the network address in IPv4 dotted decimal form, for example, `192.0.2.0`. Alternatively, set the address in IPv6 hexadecimal form, for example `2001:0DB8:0204:acff:fe97:2c34:fde0:3485`.

If the **`ping`** command fails, correct your TCP/IP configuration.

2. Set the `MQSERVER` environment variable. From the client, enter one of the following commands:
   a. On Windows:

   ```
   SET MQSERVER=channelName/TCP/server-address(port)
   ```

   b. On UNIX and Linux, and IBM WebSphere MQ client for HP Integrity NonStop Server OSS systems:

   ```
   export MQSERVER=channelName/TCP/'server-address(port)'
   ```

   c. On IBM WebSphere MQ client for HP Integrity NonStop Server Guardian systems:

   ```
   param MQSERVER CHANNEL1/TCP/server-address(port)
   ```

   d. On IBM i:

   ```
   ADDENVVAR ENVVAR(MQSERVER) VALUE('channelName/TCP/server-address(port)')
   ```

   Where:
   - *channelName* is the server-connection channel name.
   - *server-address* is the TCP/IP host name of the server.
   - *port* is the TCP/IP port number the server is listening on.

   If you do not give a port number, IBM WebSphere MQ uses the one specified in the `qm.ini` file, or the client configuration file. If no value is specified in these files, IBM WebSphere MQ uses the port number identified in the TCP/IP services file for the service name `MQSeries`. If an `MQSeries` entry in the services file does not exist, a default value of `1414` is used. It is important that the port number used by the client and the port number used by the server listener program are the same.

What to do next

Use the sample programs to test communication between the client and server; see “Testing communication between a client and a server” on page 298.

Related concepts:

“Verifying a client installation using WebSphere MQ Explorer” on page 296
You can verify a client installation using the WebSphere MQ Explorer on Windows and Linux. On the server, you create a queue manager, a local queue, a listener and a server-connection channel. On the client system you create a client-connection channel. Then from the command line you use the sample PUT and GET programs to complete the verification procedure.

“Installing a IBM WebSphere MQ client” on page 256

“Verifying a server installation” on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Related tasks:

“Setting up the server using the command line” on page 292
Follow these instructions to create a queue manager, queue, and channel on the server. You can then use these objects to verify the installation.

“Testing communication between a client and a server” on page 298
On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

Verifying a client installation using WebSphere MQ Explorer

You can verify a client installation using the WebSphere MQ Explorer on Windows and Linux. On the server, you create a queue manager, a local queue, a listener and a server-connection channel. On the client system you create a client-connection channel. Then from the command line you use the sample PUT and GET programs to complete the verification procedure.

To begin the verification setup, see "Setting up the server using WebSphere MQ Explorer."

Related concepts:

"Verifying a client installation using the command line" on page 291
You can verify a client installation using the command line. On the server you create a queue manager, a local queue, a listener, and a server-connection channel. You must also apply security rules to allow the client to connect and make use of the queue defined. On the client you create a client-connection channel, and then use the sample PUT and GET programs to complete the verification procedure.

"Installing a IBM WebSphere MQ client" on page 256

"Verifying a server installation" on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Related tasks:

"Setting up the client using WebSphere MQ Explorer" on page 297
You can use IBM WebSphere MQ Explorer to define the client-connection if you are setting up the client and server on the same workstation on a Windows or Linux system.

"Testing communication between a client and a server" on page 298
On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

Setting up the server using WebSphere MQ Explorer:

You can use the WebSphere MQ Explorer to create a queue manager, queue and server-connection channel on Windows and Linux systems .

Procedure

1. Create a queue manager:
   a. Open WebSphere MQ Explorer.
   b. Right-click the folder called **Queue Managers**, select **New** > **Queue Manager**.
   c. In the first entry field, type the queue manager name, *QUEUE.MANAGER.1*, and click **Finish**.
2. Create a local queue:
   a. Expand the queue manager you have just created and right-click **queues**.
   b. Select **New** > **Local Queue**.
   c. Enter the queue name, *QUEUE1*, and click **Finish**.
3. Define the server-connection channel:
   a. Right-click **Channels**.
   b. Select **New** > **Server Connection Channel**.
   c. Enter the channel name, *CHANNEL1*, and click **Next**.
   d. In the dialog navigation pane, click **MCA** to open the MCA page.
   e. In the MCA User ID field, enter a userid that is a member of the mqm group, typically your own.
   f. Click **Finish**.
4. Run the listener.

The listener is automatically started when the queue manager is configured. To check that the listener is running, open **Listeners** and look for `LISTENER.TCP`.

What to do next

Set up the client. See "Setting up the client using WebSphere MQ Explorer."

Related concepts:

"Verifying a client installation using the command line" on page 291
You can verify a client installation using the command line. On the server you create a queue manager, a local queue, a listener, and a server-connection channel. You must also apply security rules to allow the client to connect and make use of the queue defined. On the client you create a client-connection channel, and then use the sample PUT and GET programs to complete the verification procedure.

"Installing a IBM WebSphere MQ client" on page 256

"Verifying a server installation" on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Related tasks:

"Setting up the client using WebSphere MQ Explorer"
You can use IBM WebSphere MQ Explorer to define the client-connection if you are setting up the client and server on the same workstation on a Windows or Linux system.

"Testing communication between a client and a server" on page 298
On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

Setting up the client using WebSphere MQ Explorer:

You can use IBM WebSphere MQ Explorer to define the client-connection if you are setting up the client and server on the same workstation on a Windows or Linux system.

Procedure

1. Select the queue manager, *QUEUE.MANAGER.1*
2. Open the **Channels** folder, then right click **Client Connections** > **New** > **Client-connection Channel...**
3. Enter the channel name, *CHANNEL1*, for the client connection, and click **Next**.
4. Enter the queue manager name, *QUEUE.MANAGER.1*
5. Enter the following string as the connection name:

   ```
   server-address(port)
   ```

   Where:
   - *server-address* is the TCP/IP host name of the server
   - *port* is the TCP/IP port number the server is listening on
6. Click Finish.
7. From the command line, set the `MQCHLLIB` environment variable:
   - For Windows clients, enter the following command, where *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed:

     ```
     SET MQCHLLIB=MQ_INSTALLATION_PATH\qmgrs\QUEUE!MANAGER!1\@ipcc
     ```
   - For Linux clients, enter the following command:

     ```
     export MQCHLLIB=var/mqm/qmgrs/QUEUE!MANAGER!1/@ipcc
     ```

Note: The queue manager name contains "`.`". IBM WebSphere MQ creates the queue manager directory with the name, `QUEUE!MANAGER!1`

What to do next

Use the sample programs to test communication between the client and server. See "Testing communication between a client and a server."

Related concepts:

"Verifying a client installation using the command line" on page 291
You can verify a client installation using the command line. On the server you create a queue manager, a local queue, a listener, and a server-connection channel. You must also apply security rules to allow the client to connect and make use of the queue defined. On the client you create a client-connection channel, and then use the sample PUT and GET programs to complete the verification procedure.

"Installing a IBM WebSphere MQ client" on page 256

"Verifying a server installation" on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Related tasks:

"Testing communication between a client and a server"
On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

"Setting up the server using WebSphere MQ Explorer" on page 296
You can use the WebSphere MQ Explorer to create a queue manager, queue and server-connection channel on Windows and Linux systems .

Testing communication between a client and a server

On the IBM WebSphere MQ MQI client workstation, use the amqsputc sample program to put a message on the queue at the server workstation. Use the amqsgetc sample program to get the message from the queue back to the client.

Before you begin

Complete the previous topics in this section:
- Set up a queue manager, channels, and queue.
- Open a command window.
- Set system environment variables.

About this task

Note that IBM WebSphere MQ object definitions are case-sensitive. Text entered as an MQSC command in lowercase is converted automatically to uppercase unless you enclose it in single quotation marks. Make sure that you type the examples exactly as shown.

Procedure

1. On UNIX and Linux systems, change into the `*MQ_INSTALLATION_PATH*/samp/bin directory`, which contains the sample programs. For IBM WebSphere MQ client for HP Integrity NonStop Server, change into the `*MQ_INSTALLATION_PATH*/opt/mqm/samp/bin` directory, which contains the sample programs. `*MQ_INSTALLATION_PATH*` represents the high-level directory in which IBM WebSphere MQ is installed.
2. If you are verifying an installation on a UNIX, Linux, or Windows system, you must set certain environment variables so that the installation can be used in the current shell. This step is not

applicable to IBM WebSphere MQ client for HP Integrity NonStop Server. You can set the environment variables by entering one of the following commands:

- Windows:

  ```
  MQ_INSTALLATION_PATH\bin\setmqenv -s
  ```

  where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
- UNIX and Linux:

  ```
  . MQ_INSTALLATION_PATH/bin/setmqenv -s
  ```

  where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

3. Start the PUT program for `QUEUE1` on `QUEUE.MANAGER.1` by entering one of the following commands:
   - Windows:

     ```
     amqsputc QUEUE1 QUEUE.MANAGER.1
     ```
   - UNIX and Linux, and IBM WebSphere MQ client for HP Integrity NonStop Server:

     ```
     ./amqsputc QUEUE1 QUEUE.MANAGER.1
     ```

   If the command is successful, the following messages are displayed:

   ```
   Sample AMQSPUT0 start target queue is QUEUE1
   ```

   Tip: You might get the error, `MQRC_NOT_AUTHORIZED` (2035). By default, channel authentication is enabled when a queue manager is created. Channel authentication prevents privileged users accessing a queue manager as a IBM WebSphere MQ MQI client. For verifying the installation, you can either change the MCA user ID to a non-privileged user, or disable channel authentication. To disable channel authentication run the following MQSC command:

   ```
   ALTER QMGR CHLAUTH(DISABLED)
   ```

   When you finish the test, if you do not delete the queue manager, re-enable channel authentication:

   ```
   ALTER QMGR CHLAUTH(ENABLED)
   ```
4. Type some message text, then press **Enter** twice. The following message is displayed:

   ```
   Sample AMQSPUT0 end
   ```

   Your message is now on the queue that is on the server queue manager.
5. Start the GET program for `QUEUE1` on `QUEUE.MANAGER.1` by entering one of the following commands:
   - Windows:

     ```
     amqsgetc QUEUE1 QUEUE.MANAGER.1
     ```
   - UNIX and Linux, and IBM WebSphere MQ client for HP Integrity NonStop Server:

     ```
     ./amqsgetc QUEUE1 QUEUE.MANAGER.1
     ```

   The sample program starts, and your message is displayed. After a short pause (approximately 30 seconds), the sample ends and the command prompt is displayed again.

Results

You have now successfully verified the client installation.

What to do next

1. If your server is a Windows, UNIX, or Linux system, you must set various environment variables so that the installation can be used in the current shell. You can set the environment variables by entering one of the following commands:
   - Windows:

     ```
     MQ_INSTALLATION_PATH\bin\setmqenv -s
     ```

where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

- UNIX and Linux systems:

  ```
  . MQ_INSTALLATION_PATH/bin/setmqenv -s
  ```

  where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

2. On the server, stop the queue manager by entering the following command:

   ```
   endmqm QUEUE.MANAGER.1
   ```

3. On the server, delete the queue manager by entering the following command:

   ```
   dltmqm QUEUE.MANAGER.1
   ```

Related concepts:

"Verifying a client installation using the command line" on page 291
You can verify a client installation using the command line. On the server you create a queue manager, a local queue, a listener, and a server-connection channel. You must also apply security rules to allow the client to connect and make use of the queue defined. On the client you create a client-connection channel, and then use the sample PUT and GET programs to complete the verification procedure.

"Verifying a client installation using WebSphere MQ Explorer" on page 296
You can verify a client installation using the WebSphere MQ Explorer on Windows and Linux. On the server, you create a queue manager, a local queue, a listener and a server-connection channel. On the client system you create a client-connection channel. Then from the command line you use the sample PUT and GET programs to complete the verification procedure.

"Installing a IBM WebSphere MQ client" on page 256

"Verifying a server installation" on page 280
You can verify a local (stand-alone) installation or a server-to-server installation of the IBM WebSphere MQ server. A local installation has no communication links with other IBM WebSphere MQ installations while a server-to-server installation does have links to other installations.

Related tasks:

"Installing a WebSphere MQ server" on page 209
After preparing your system for installation you may install IBM WebSphere MQ by following the appropriate instructions for your platform. After installation, you might want to verify your installation to check that installation has been successful.

Verifying the installation of IBM WebSphere MQ Telemetry

There are two ways to verify the installation of IBM WebSphere MQ Telemetry. Either one or both can be used, regardless of whether IBM WebSphere MQ Telemetry was installed as a custom installation of IBM WebSphere MQ, or added to an existing installation of IBM WebSphere MQ.

The two ways to verify the installation of IBM WebSphere MQ Telemetry are:

- "Verifying the installation of IBM WebSphere MQ Telemetry by using IBM WebSphere MQ Explorer"
- "Verifying the installation of IBM WebSphere MQ Telemetry using the command line" on page 302

Verifying the installation of IBM WebSphere MQ Telemetry by using IBM WebSphere MQ Explorer

Use the Define sample configuration wizard and the MQTT client utility in IBM WebSphere MQ Explorer to verify that the IBM WebSphere MQ Telemetry components have installed. Also check that publish/subscribe works correctly.

Before you begin

The IBM WebSphere MQ Telemetry runtime and support for IBM WebSphere MQ Explorer must be installed. The telemetry folder is part of a queue manager. To view the telemetry folder, you must start a queue manager.

Before running the define sample configuration wizard on an existing queue manager, review the information provided by the wizard about the configuration changes that are made. The changes might have implications for the configuration of the existing queue manager. Alternatively, run the sample configuration wizard on a newly created queue manager to avoid changing any security settings.

About this task

To configure IBM WebSphere MQ Telemetry there is a define sample configuration wizard that can be run from IBM WebSphere MQ Explorer. The wizard runs through a series of steps, including defining and starting the telemetry (MQXR) service, setting up the default transmission queue, and configuring a telemetry channel.

If you would prefer to do this manually, see Configuring a queue manager for telemetry on Linux and AIX. For Windows, see Configuring a queue manager for telemetry on Windows.

You can open the define sample configuration wizard from the IBM WebSphere MQ Telemetry Welcome page in IBM WebSphere MQ Explorer. The wizard determines which steps are required based on the current configuration.

For example, the following actions might be specified by the wizard:

- Define the telemetry (MQXR) service.
- Start the telemetry (MQXR) service.
- Define the telemetry transmit queue.
- Set the default transmit queue of the queue manager to `SYSTEM.MQTT.TRANSMIT.QUEUE`.

If telemetry is already configured for this queue manager, the link to open the wizard is replaced with static text. The text confirms that the sample configuration has been set up.

After the configuration has finished, you can use IBM WebSphere MQ Explorer to open the MQTT client utility. Use the MQTT client utility to verify that IBM WebSphere MQ Telemetry is set up correctly.

The following items summarize the main goals that can be achieved using the MQTT client utility:

- Validation of a basic or custom IBM WebSphere MQ Telemetry configuration by connecting, subscribing to topics and publishing messages.
- Showcases the main features of MQTT protocol.
- Provides a simple tool to aid in debugging IBM WebSphere MQ Telemetry applications.

You can find additional information within the IBM WebSphere MQ Explorer by using the **Help** menu or pressing the **F1** key.

Procedure

1. Start IBM WebSphere MQ Explorer.

   On Windows and Linux systems, you can start IBM WebSphere MQ Explorer by using the system menu, the `MQExplorer` executable file, the **`mqexplorer`** command, or the **`strmqcfg`** command.
2. Open the Welcome to MQ Telemetry page.
   - To use an existing queue manager, click on `IBM WebSphere MQ\Queue Managers\`*`qMgrName`*`\Telemetry` folder to open the Welcome to MQ Telemetry page.
   - If, for the reasons mentioned, you decide to use a new queue manager,
     a. Click **Queue Managers** > **New** > **Queue Manager**.
     b. Type `MQTTVerification` as the **Queue manager name** > **Next** > **Next** > **Next**.
     c. Change the default port in **Listen on port number**, if the port is in use > **Finish**.

   d. When the queue manager starts, click on `IBM WebSphere MQ\Queue Managers\MQTTVerification\Telemetry` folder to open the Welcome to MQ Telemetry page.

3. From the Welcome to MQ Telemetry page in IBM WebSphere MQ Explorer, click **Define sample configuration**.

   If this link is not present, and instead you see the text, “The sample configuration has been set up for this queue manager”, then telemetry has already been configured. Proceed to step 6.

   If you clicked **Define sample configuration**, the page opens, and lists actions that are to be performed as part of the sample configuration.
4. Leave **Launch MQTT client utility** checked, if you want to automatically start the MQTT client utility. The check box is selected by default.
5. Click **Finish**.
6. Click **Connect**.

   In the MQTT client utility panel, ensure that the host and port names are correct.

   If you did not automatically start the MQTT client utility panel in step 4, you can start it either by using a direct link from the Welcome to MQ Telemetry panel, or by right-clicking a NON-SSL channel, which allows you to control the channel it runs on.

   The client history records a `Connected` event.
7. Click **Subscribe**.

   The client history records a `Subscribed` event.
8. Click **Publish**.

   The client history records a `Published` and `Received` event.

Results

If the publish/subscribe finishes successfully, the IBM WebSphere MQ Telemetry installation is verified.

If you encounter problems during the installation process, view the error log:
- On Windows, the default location for this log is, *WebSphere MQ data directory*`\qmgrs\`*qMgrName*`\mqxr`
- On AIX and Linux, the default location for this log is, `/var/mqm/qmgrs/`*qMgrName*`/mqxr/`

Verifying the installation of IBM WebSphere MQ Telemetry using the command line

Follow these instructions to run scripts to verify that the IBM WebSphere MQ Telemetry components have installed, and are able to publish and subscribe.

Before you begin

The telemetry (MQXR) service must be started to run the sample programs. The user ID must be a member of the mqm group. You must run the SampleMQM script first, followed by the MQTTV3Sample script to perform a publish and subscribe. The CleanupMQM sample script can be run to delete the queue manager created by the SampleMQM script. There are two commands to run the MQTTV3Sample script. These commands are shown in the Procedure section of this topic. The commands must be entered into different command lines or shell windows.

The SampleMQM script creates and uses a queue manager called MQXR_SAMPLE_QM. Therefore, do not run unaltered on a system with a MQXR_SAMPLE_QM queue manager. Any changes made might have implications for the configuration of the existing queue manager.

About this task

To perform verification on a server or device without a GUI, scripts are provided in the samples directory. The SampleMQM script performs the required steps to configure IBM WebSphere MQ

Telemetry. The SampleMQM script runs through a series of steps, including defining and starting the telemetry (MQXR) service, setting up the default transmission queue, and configuring a telemetry channel.

For information about performing this manually, see Configuring a queue manager for telemetry on Linux and AIX , or Configuring a queue manager for telemetry on Windows.

The MQTTV3Sample script can then be run to validate the basic or custom IBM WebSphere MQ Telemetry configuration by connecting, subscribing to topics, and publishing messages.

The following items summarize the main goals that can be achieved using the verification scripts provided.

- Validation of a basic or custom IBM WebSphere MQ Telemetry configuration by connecting, subscribing to topics and publishing messages.
- Showcases the main features of MQTT protocol.
- Provides a simple tool to aid in debugging IBM WebSphere MQ Telemetry applications.

Procedure

1. Type the following command on a command line to run the SampleMQM script
   - On Windows, the command to run the SampleMQM script is as follows:
     ```
     MQ_INSTALLATION_PATH\mqxr\samples\SampleMQM.bat
     ```
   - On AIX and Linux, the command to run the SampleMQM script is as follows:
     ```
     MQ_INSTALLATION_PATH/mqxr/samples/SampleMQM.sh
     ```

   where *MQ_INSTALLATION_PATH* is the location in which IBM WebSphere MQ is installed.

   A queue manager called MQXR_SAMPLE_QM is created.
2. Type the following command to run the first part of the MQTTV3Sample script;
   - On Windows, on one command line, type the following command;
     ```
     MQ_INSTALLATION_PATH\mqxr\samples\RunMQTTV3Sample.bat -a subscribe
     ```
   - On AIX and Linux, in one shell window, type the following command;
     ```
     MQ_INSTALLATION_PATH/mqxr/samples/RunMQTTV3Sample.sh -a subscribe
     ```
3. Type the following command to run second part of the MQTTV3Sample script;
   - On Windows, on another command line, type the following command;
     ```
     MQ_INSTALLATION_PATH\mqxr\samples\RunMQTTV3Sample.bat -m "Hello from an MQTT v3 application"
     ```
   - On AIX and Linux, in another shell window, type the following command;
     ```
     MQ_INSTALLATION_PATH/mqxr/samples/RunMQTTV3Sample.sh -m "Hello from an MQTT v3 application"
     ```
4. To remove the queue manager created by the SampleMQM script, you can run the CleanupMQM script using the following command;
   - On Windows, type the following command;
     ```
     MQ_INSTALLATION_PATH\mqxr\samples\CleanupMQM.bat
     ```
   - On AIX and Linux in another shell window, type the following command;
     ```
     MQ_INSTALLATION_PATH/mqxr/samples/CleanupMQM.sh
     ```

Results

The `Hello from an MQTT v3 application` message, that you typed into the second command line or window, is published by that application, and received by the application in the first window. The application in the first window shows it on the screen.

Press **Enter** in the first window to end the subscribing application.

If the scripts finish and messages can be sent and received, the IBM WebSphere MQ Telemetry installation is verified.

If you encounter any problems during the installation process, see Troubleshooting. Alternatively, you can view the Error Log.

- On Windows operating systems, the default location for the log is created in *WMQinstallation*\qmgrs*queue manager name*\mqxr
- On AIX and Linux systems, the default location for this log is created in /var/mqm/qmgrs/*queue manager name*/mqxr/

where *queue manager name* is the name of the queue manager.

Uninstalling IBM WebSphere MQ components

Information to help you uninstall components of IBM WebSphere MQ.

About this task

Refer to the following topics to uninstall components of IBM WebSphere MQ:

- Uninstalling IBM WebSphere MQ server and client
- Uninstalling IBM WebSphere MQ Telemetry
- Uninstalling IBM WebSphere MQ Advanced Message Security

Uninstalling

The topics in this section provide instructions on how to uninstall components.

Select the appropriate topic for your platform to find out how to uninstall WebSphere MQ components:

- "Uninstalling IBM WebSphere MQ on AIX"
- "Uninstalling IBM WebSphere MQ on HP-UX" on page 306
- "Uninstalling IBM WebSphere MQ on Linux" on page 307
- "Uninstalling IBM WebSphere MQ on Solaris" on page 308
- "Uninstalling IBM WebSphere MQ on Windows systems" on page 309

Related concepts:

"Installing IBM WebSphere MQ" on page 209
The topics in this section provide instructions on how to install IBM WebSphere MQ.

"Verifying a WebSphere MQ installation" on page 280
The topics in this section provide instructions on how to verify a server or a client installation of IBM WebSphere MQ on Windows, UNIX and Linux, and HP Integrity NonStop Server systems.

Uninstalling IBM WebSphere MQ on AIX

On AIX, you can uninstall theIBM WebSphere MQ server or client using the System Management Interface Tool (SMIT) or the **`installp`** command.

Before you begin

If any updates have been applied, remove them before starting this uninstallation procedure. For more information, see AIX: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5 .

Procedure

1. Stop all IBM WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any IBM WebSphere MQ activity associated with the installation you are uninstalling:

a. Log in as a user in the group mqm.

b. Set up your environment to work with the installation you want to uninstall. Enter the following command:

```
. MQ_INSTALLATION_PATH/bin/setmqenv
```

where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

c. Display the state of all queue managers on the system. Enter the following command:

```
dspmq -o installation
```

d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:

```
endmqm QMgrName
```

e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:

```
endmqlsr -m QMgrName
```

3. Log in as root.
4. Uninstall IBM WebSphere MQ using either **installp** or **smit**. If IBM WebSphere MQ was installed in a non-default location, you must use **installp** to uninstall.
   - Uninstall using **installp** by entering one of the following commands:
     - For an installation in the default location /usr/mqm

       ```
       installp -u mqm
       ```

     - For an installation in a non-default location:

       ```
       installp -R usil -u mqm
       ```

       where *usil* is the path of the User Specified Install Location (USIL) specified when the product was installed.
   - Uninstall using **smit**:

     a. Select the required **smit** window using the following sequence:

     ```
     Software Installation and Maintenance
       Software Maintenance and Utilities
         Remove Installed Software
     ```

     b. List the software in the **SOFTWARE name** field:

     1) Enter **.**

     2) Press **F4**

     c. Select the file sets to uninstall from the list (those beginning with mqm), and press **Enter**. There is an option at this stage to do a preview. Leave the option set to the default value of **Yes** to preview the file sets you are uninstalling, or select **No** to not preview these file sets.

     d. Press **Enter** on the **Remove Installed Software** panel, it asks whether you are sure, press **Enter**.

Results

You might find that certain files remain after uninstalling IBM WebSphere MQ. For example, mqinst.ini remains after uninstallation. This is to enable information to be reused by subsequent installs. See below for more information.

What to do next

- If the product successfully uninstalled, you can delete any files and directories contained in the /usr/mqm directory under the User Specified Install Location (USIL) specified in the **installp** uninstallation command.
- Use the **lslpp** command to check for other products installed in the USIL. If there are no other products installed in the USIL and you do not intend to use it again, you can delete the USIL using the **rmusil** command.

- If there are no other IBM WebSphere MQ installations on the system, and you are not planning to reinstall or migrate, you can delete the /var/mqm and /etc/opt/mqm directory trees. Deleting these directories destroys all queue managers and their associated data.

Uninstalling IBM WebSphere MQ on HP Integrity NonStop Server

On HP Integrity NonStop Server systems, you can uninstall the IBM WebSphere MQ client by using the **rm** command.

Procedure

1. Stop all IBM WebSphere MQ applications that are associated with the installation you are uninstalling.
2. Log in to the OSS as the user ID that owns the installation.
3. Use the OSS **rm** command to delete the files from the Guardian subvolume used by the installation. For example, use the following command:
   ```
   rm -rf <mqpath>/opt/mqm/bin/G/*
   ```
4. Use the OSS **rm** command to delete the OSS directory trees for the installation. For example, use the following command:
   ```
   rm -rf <mqpath>
   ```

Uninstalling IBM WebSphere MQ on HP-UX

On HP-UX, you can uninstall theIBM WebSphere MQ server or client using the **swremove** command.

Before you begin

If any updates have been applied, remove them before starting this uninstallation procedure. For more information, see HP-UX: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5 .

Procedure

1. Stop all IBM WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any IBM WebSphere MQ activity associated with the installation you are uninstalling:
   a. Log in as a user in the group mqm.
   b. Set up your environment to work with the installation you want to uninstall. Enter the following command:
   ```
   . MQ_INSTALLATION_PATH/bin/setmqenv
   ```
   where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
   c. Display the state of all queue managers on the system. Enter the following command:
   ```
   dspmq -o installation
   ```
   d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:
   ```
   endmqm QMgrName
   ```
   e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:
   ```
    endmqlsr -m QMgrName
   ```
3. Log in as root.
4. Uninstall IBM WebSphere MQ using **swremove**:
   - To uninstall all IBM WebSphere MQ components, enter the following command:
     ```
     swremove MQSERIES,l=MQ_INSTALLATION_PATH
     ```
     where *MQ_INSTALLATION_PATH* is the path where IBM WebSphere MQ is installed.
   - To uninstall selected IBM WebSphere MQ components, enter the following command:

```
swremove componentname,l=MQ_INSTALLATION_PATH
```

where *componentname* is the name of the component to uninstall, and *MQ_INSTALLATION_PATH* is the path where IBM WebSphere MQ is installed.

For example, to uninstall the client and the telemetry client components, from an installation in /opt/myLocation, enter the following command:

```
swremove MQSERIES.MQM-TXCLIENT,l=/opt/myLocation MQSERIES.MQM-CL-HPUX,l=/opt/myLocation
```

Results

You might find that certain files remain after uninstalling IBM WebSphere MQ. For example, mqinst.ini remains after uninstallation. This is to enable information to be reused by subsequent installs. See below for more information.

What to do next

- If the product successfully uninstalled, you can delete any files and directories contained in the installation directory.
- If there are no other IBM WebSphere MQ installations on the system, and you are not planning to reinstall or migrate, you can delete the /var/mqm and /etc/opt/mqm directory trees. Deleting these directories destroys all queue managers and their associated data.

Uninstalling IBM WebSphere MQ on Linux

On Linux, you can uninstall theIBM WebSphere MQ server or client using the **rpm** command.

Before you begin

If any updates have been applied, remove them before starting this uninstallation procedure. For more information, see Linux: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5 .

Procedure

1. Stop all IBM WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any IBM WebSphere MQ activity associated with the installation you are uninstalling:
   a. Log in as a user in the group mqm.
   b. Set up your environment to work with the installation you want to uninstall. Enter the following command:
      ```
      . MQ_INSTALLATION_PATH/bin/setmqenv -s
      ```
      where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.
   c. Display the state of all queue managers on the system. Enter the following command:
      ```
      dspmq -o installation
      ```
   d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:
      ```
      endmqm QMgrName
      ```
   e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:
      ```
       endmqlsr -m QMgrName
      ```
3. Log in as root.
4. Uninstall IBM WebSphere MQ using the **rpm** command:
   a. On a system with a single installation:
      1) Find out the names of the packages (components) currently installed on your system, by entering the following command:

```
rpm -qa | grep MQSeries
```

2) Remove all the components at the same time by appending all the package names to the **rpm** command arguments. For example, to remove the runtime, Server and SDK components enter the following command:

```
rpm -ev MQSeriesRuntime MQSeriesServer MQSeriesSDK
```

Tip: To list the packages, and uninstall them in one go, use a command like the following one:

```
rpm -qa | grep MQSeries | xargs rpm -ev
```

b. On a system with multiple installations:

1) Find out the names of the packages (components) currently installed on your system, by entering the following command:

```
rpm -qa | grep suffix
```

where *suffix* is the unique name given to the packages when **crtmqpkg** was run at installation time. *suffix* is included in each of the package names that belong to a particular installation.

2) Remove all the components at the same time by appending all the package names to the **rpm** command arguments. For example, to remove the runtime, Server and SDK components for an installation with the *suffix* MQ71, enter the following command:

```
rpm -ev MQSeriesRuntime-MQ71 MQSeriesServer-MQ71 MQSeriesSDK-MQ71
```

Results

You might find that certain files remain after uninstalling IBM WebSphere MQ. For example, mqinst.ini remains after uninstallation. This is to enable information to be reused by subsequent installs. See below for more information.

What to do next

- If the product successfully uninstalled, you can delete any files and directories contained in the installation directory.
- If there are no other IBM WebSphere MQ installations on the system, and you are not planning to reinstall or migrate, you can delete the /var/mqm and /etc/opt/mqm directory trees. Deleting these directories destroys all queue managers and their associated data.

Uninstalling IBM WebSphere MQ on Solaris

On Solaris, you can uninstall theIBM WebSphere MQ server or client using the **pkgrm** command.

Before you begin

If any updates have been applied, remove them before starting this uninstallation procedure. For more information, see Solaris: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5 .

Procedure

1. Stop all IBM WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any IBM WebSphere MQ activity associated with the installation you are uninstalling:

a. Log in as a user in the group mqm.

b. Set up your environment to work with the installation you want to uninstall. Enter the following command:

```
. MQ_INSTALLATION_PATH/bin/setmqenv
```

where *MQ_INSTALLATION_PATH* refers to the location where IBM WebSphere MQ is installed.

c. Display the state of all queue managers on the system. Enter the following command:

```
dspmq
```

d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:

```
endmqm QMgrName
```

e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:

```
 endmqlsr -m QMgrName
```

3. Log in as root.
4. Uninstall IBM WebSphere MQ using **pkgrm**:

a. On a system with a single installation, enter the following command:

```
pkgrm mqm
```

b. On a system with multiple installations:

```
pkgrm mqm-suffix
```

where *suffix* is the unique name given to the packages when **crtmqpkg** was run at installation time. *suffix* is included in each of the package names that belong to a particular installation. The first installation on the system does not have a *suffix*, and is uninstalled using the same method as for a single installation.

If a package has a dependency on mqm, **pkgrm** returns the name of the package. Uninstall the dependent packages first.

Results

You might find that certain files remain after uninstalling IBM WebSphere MQ. For example, mqinst.ini remains after uninstallation. This is to enable information to be reused by subsequent installs. See below for more information.

What to do next

- If the product successfully uninstalled, you can delete any files and directories contained in the installation directory.
- If there are no other IBM WebSphere MQ installations on the system, and you are not planning to reinstall or migrate, you can delete the /var/mqm and /etc/opt/mqm directory trees. Deleting these directories destroys all queue managers and their associated data.

Uninstalling IBM WebSphere MQ on Windows systems

You can uninstall the IBM WebSphere MQ MQI clients and servers on Windows systems by using the control panel, the command line (**msiexec**), **MQParms**, or by using the installation media, in which case you can optionally remove queue managers as well.

Before you begin

By default, uninstall logging is not enabled on Windows. To ensure that you receive an uninstall log, carry out the following procedure:

1. In a command prompt, open the registry editor by issuing the command **regedit**.
2. Go to the appropriate registry key: HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer
3. Under this registry key add the following information:

Name Logging

Data type
REG_SZ

Value voicewarmup

4. Save the updated registry key.

Procedure

The first part of the procedure ensures that there are no IBM WebSphere MQ programs or processes running:

1. If you are running IBM WebSphere MQ with the Microsoft Cluster Service (MSCS), remove the queue managers from MSCS control before uninstalling IBM WebSphere MQ. Perform the following steps for each queue manager currently under MSCS control :
   a. Take the queue manager resource offline.
   b. Destroy the resource instance.
   c. Migrate the queue manager files back from shared drives. This step is shown as optional in Removing a queue manager from MSCS control. However, it is mandatory in this case.
2. Stop all IBM WebSphere MQ applications associated with the installation you are uninstalling.
3. Close all IBM WebSphere MQ Managed File Transfer agents. If you have a IBM WebSphere MQ Managed File Transfer agent running, close it by using the **fteStopAgent** command; see fteStopAgent (stop a IBM WebSphere MQ Managed File Transfer agent).
4. For a server installation, end all IBM WebSphere MQ activity:
   a. Log in as a user in the group mqm.
   b. Stop all running queue managers and listeners by using the IBM WebSphere MQ Explorer, or by entering the following commands:
      1) Set up your environment to work with the installation you want to uninstall by entering the following command:
      ```
      MQ_INSTALLATION_PATH\bin\setmqenv -s
      ```
      where *MQ_INSTALLATION_PATH* is the location where IBM WebSphere MQ is installed.
      2) For each queue manager, enter the following command to stop the queue manager:
      ```
      endmqm queue_manager_name
      ```
      3) For each queue manager, enter the following command to stop any listeners associated with the queue manager:
      ```
       endmqlsr -m queue_manager_name
      ```
5. Stop IBM WebSphere MQ. To do this right-click the **WebSphere MQ** icon in the system tray, then select **Stop WebSphere MQ**.
6. Close all IBM WebSphere MQ windows.
7. Stop any monitoring service.

When all processes associated with IBM WebSphere MQ are no longer running, you can uninstall IBM WebSphere MQ:

8. Uninstall IBM WebSphere MQ by using one of the following methods:
   - Use the Windows Control Panel. This process is described in: "Uninstalling IBM WebSphere MQ by using the control panel" on page 311. This method does not remove the queue manager data.
   - Use the command line by running the **msiexec** command as described in: "Uninstalling IBM WebSphere MQ by using the command line" on page 312. This method does not remove the queue manager data.
   - Use the appropriate parameters with **MQParms**. This process is described in "Uninstalling IBM WebSphere MQ by using MQParms" on page 313. This method does not remove the queue manager data.
   - Use the installation media, by selecting the appropriate option as described in: "Uninstalling IBM WebSphere MQ on Windows by using the installation media" on page 314. The option to remove queue manager data is displayed in the Removing Server feature panel, if appropriate.

If you have to cancel the uninstall process before it is finished, you might have to reconfigure IBM WebSphere MQ with the Prepare IBM WebSphere MQ wizard because the rollback of the deletion of the IBM WebSphere MQ service is unable to set the service's user account password. Use the following command to reconfigure IBM WebSphere MQ:

```
MQ_INSTALLATION_PATH\bin\amqmjpse.exe -r
```

For more information about the Prepare IBM WebSphere MQ wizard, see "Configuring WebSphere MQ with the Prepare WebSphere MQ wizard" on page 246.

9. Check the Windows event log and restart the system if necessary. If event ID 10005 is written to the Windows event log, you must restart the system to complete the uninstall.
10. Optional: If you are uninstalling the last or only installation of IBM WebSphere MQ, you can remove all the information about previous installations that is retained on the system, if you want to.

    Two registry values remain after uninstallation:

    - 32 bit systems:
      - `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere MQ\LogDefaultPath`
      - `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere MQ\WorkPath`
    - 64 bit systems:
      - `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\IBM\WebSphere MQ\LogDefaultPath`
      - `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\IBM\WebSphere MQ\WorkPath`

    Data folders are also left intact, located at *`MQ_DATA_PATH`*`\Config`, where *`MQ_DATA_PATH`* is the location of the IBM WebSphere MQ data directory.

    If a client is installed on a system where the `LogDefaultPath` registry value remains from a previous server install, a client installation will attempt to create this directory if it does not already exist. If this behavior is not wanted, remove the LogDefaultPath registry value before installing the client.

Uninstalling IBM WebSphere MQ by using the control panel:

You can uninstall IBM WebSphere MQ by using the control panel to remove all currently installed features.

Before you begin

Start the uninstalling process by following the steps described in "Uninstalling IBM WebSphere MQ on Windows systems" on page 309.

If you no longer require the queue managers that are on the system, delete them by using the IBM WebSphere MQ Explorer or the **`dltmqm`** command.

Procedure

1. From the Windows taskbar, open the control panel by clicking **Start** > **Settings** > **Control Panel**, or **Start** > **Control Panel**.
2. Click **Add/Remove Programs**, or **Programs and Features**.
3. Click **IBM WebSphere MQ (*installation_name*)**, where *installation_name* is the name of the installation you want to remove.
4. Click **Remove** or **Uninstall** and click **Yes** to confirm. If User Account Control (UAC) is enabled, accept the Windows prompt to allow the uninstallation to run as elevated. The program then begins and runs to completion.

What to do next

Complete the steps that you started in "Uninstalling IBM WebSphere MQ on Windows systems" on page 309.

Uninstalling IBM WebSphere MQ by using the command line:

You can uninstall IBM WebSphere MQ by running the **msiexec** command from the command line to remove all currently installed features.

Before you begin

Start the uninstalling process by following the steps described in "Uninstalling IBM WebSphere MQ on Windows systems" on page 309.

If you no longer require the queue managers that are on the system, delete them by using the IBM WebSphere MQ Explorer or the **dltmqm** command.

About this task

To start uninstalling, use the **msiexec** command.

If you are running IBM WebSphere MQ on Windows with User Account Control (UAC) enabled, you must invoke the silent uninstallation from an elevated command prompt. Elevate a command prompt by using a right-click to start the command prompt and choose **Run as administrator**.

In all of the examples of commands shown, the variable names used are as follows:

- *installation_name* is the name of the installation you want to remove.
- *product_code* is the value shown for MSIProdCode in the output of the following command:

  ```
  dspmqinst -n installation_name
  ```

  An example of a product code is {0730749B-080D-4A2E-B63D-85CF09AE0EF0}.
- *response_file* is the file that contains the [Response] stanza and the required *property*=*value* pairs. For details about how to create a response file, see "Creating a response file" on page 238. For details of the parameters you can specify in a response file, see Table 39 on page 234 in "Advanced installation using msiexec" on page 231. This is an example of a simple uninstallation [Response] stanza:

  ```
  [Response] REMOVE="ALL"
  ```

Procedure

To uninstall all IBM WebSphere MQ features, use one of the following methods:

- Run the msiexec command with a parameter that calls a response file.

  A response file is an ASCII text file that contains the parameter values that you want to set for the uninstallation. The response file has a format similar to a Windows .ini file, and contains the stanza [Response]. This stanza contains parameters that the **msiexec** command can use, in the form of *property*=*value* pairs. The **msiexec** command ignores any other stanzas in the file.

  You can set which features to uninstall, and set whether to keep existing queue managers.

  To silently uninstall IBM WebSphere MQ using a response file, enter the following command:

  ```
  msiexec /i {product_code} /l*v "c:\removal.log" /q USEINI="response_file" INSTALLATIONNAME="installation_name"
  ```
- Enter one of the following commands on the command line:
  - To invoke an interactive uninstallation giving you the option to remove queue manager data (providing there are no other IBM WebSphere MQ installations remaining):

    ```
    msiexec /i {product_code} /l*v "c:\removal.log" REMOVE="All" INSTALLATIONNAME="installation_name"
    ```

    If you are running IBM WebSphere MQ on a Windows system with User Account Control (UAC) enabled, you might see Open File - Security Warning dialog boxes during uninstallation that list International Business Machines Limited as the publisher. Click **Run** to allow the uninstallation to continue.
  - To invoke a silent uninstallation that does not remove any queue manager data:

```
msiexec /i {product_code} /l*v "c:\removal.log" /q REMOVE="All" INSTALLATIONNAME="installation_name"
```

- To invoke a silent uninstallation and remove any queue manager data (only valid when removing the final server installation):

  ```
  msiexec /i {product_code} /l*v "c:\removal.log" /q REMOVE="All" KEEPQMDATA="delete" INSTALLATIONNAME="installation_
  ```

- To the progress of the uninstalling process and not remove any queue manager data:

  ```
  msiexec /x {product_code} /l*v "c:\removal.log" INSTALLATIONNAME="installation_name"
  ```

  If you are running IBM WebSphere MQ on a Windows system with User Account Control (UAC) enabled, you might see Open File - Security Warning dialog boxes during uninstallation that list International Business Machines Limited as the publisher. Click **Run** to allow the uninstallation to continue.

- To invoke a silent uninstallation and not remove any queue manager data:

  ```
  msiexec /x {product_code} /l*v "c:\removal.log" /q INSTALLATIONNAME="installation_name"
  ```

Results

After the command is entered, the command prompt immediately reappears and IBM WebSphere MQ is uninstalled as a background process. If you entered parameters to produce a log, check this file to see how the uninstallation is progressing. If the uninstallation finishes successfully, you see the message “Removal completed successfully” in the log file.

What to do next

Complete the steps that you started in “Uninstalling IBM WebSphere MQ on Windows systems” on page 309.

Uninstalling IBM WebSphere MQ by using MQParms:

You can uninstall IBM WebSphere MQ by running the **MQParms** command from the command line to remove all currently installed features.

Before you begin

Start the uninstalling process by following the steps described in “Uninstalling IBM WebSphere MQ on Windows systems” on page 309.

Procedure

1. Follow the instructions on the MQParms installation pages to uninstall IBM WebSphere MQ non-interactively. See: “Using the MQParms command” on page 239.
   a. Set the ADDLOCAL parameter to empty (ADDLOCAL="").
   b. Set the REMOVE parameter to "ALL" (REMOVE="ALL").
2. If you have multiple versions of IBM WebSphere MQ installed on your system, specify the product code that identifies the installation you want to remove. Type the following command:

   ```
   MQParms.exe parameter_file /i "{product_code}"
   ```

   where

   - *parameter_file* is the file that contains the required parameter values. If this file is not in the same folder as MQParms.exe, specify the full path and file name. If you do not specify a parameter file, the default is MQParms.ini.
   - *product_code* is the value shown for MSIProdCode in the output of the following command:

     ```
     dspmqinst -n installation_name
     ```

     where *installation_name* is the name of the installation you want to remove. An example of a product code is {0730749B-080D-4A2E-B63D-85CF09AE0EF0}.

What to do next

Complete the steps that you started in "Uninstalling IBM WebSphere MQ on Windows systems" on page 309.

Uninstalling IBM WebSphere MQ on Windows by using the installation media:

You can uninstall IBM WebSphere MQ by using the installation media to remove all currently installed features and optionally remove existing queue managers and their data.

Before you begin

Start the uninstalling process by following the steps described in "Uninstalling IBM WebSphere MQ on Windows systems" on page 309.

Procedure

1. Insert the IBM WebSphere MQ for Windows Server DVD into the DVD drive.
2. Start the installation process.
   - If autorun is enabled, the installation process starts automatically.
   - If autorun is not enabled, double-click the **Setup** icon in the root folder of the DVD to start the installation process.

   The IBM WebSphere MQ Installation Launchpad window opens.
3. Click **WebSphere MQ Installation**.
4. Click **Launch IBM WebSphere MQ Installer** and click **Next** until the IBM WebSphere MQ Program Maintenance panel is displayed with a welcome message. If this panel is not displayed, IBM WebSphere MQ for Windows is not currently installed.
5. Click **Maintain or upgrade an existing instance** and if there is more than one installation of IBM WebSphere MQ on the system, select which installation you want to remove. Click **Next** and in the Program Maintenance panel, click **Remove**, then **Next**.
6. If you are uninstalling the last or only server, and there are any queue managers on the system, the Removing Server feature panel is shown. Click one of the following options:
   - **Keep**: keep existing queue managers and their objects.
   - **Remove**: remove existing queue managers and their objects.

   Click **Next**. The Remove IBM WebSphere MQ panel is displayed, with a summary of the installation to be removed.
7. Click **Remove** to continue. If there are any messages that state that locked files are found, ensure that there are no IBM WebSphere MQ programs running; see "Uninstalling IBM WebSphere MQ on Windows systems" on page 309. When IBM WebSphere MQ has been uninstalled, a message indicates completion.
8. Click **Finish**.

What to do next

Complete the steps that you started in "Uninstalling IBM WebSphere MQ on Windows systems" on page 309.

Uninstalling IBM WebSphere MQ Telemetry components

Remove IBM WebSphere MQ Telemetry components from your computer.

Before you begin

Check that your user ID has the following authority to complete uninstallation tasks:

- On all Windows operating systems and editions, your user ID must be a member of the Administrators group.
- On AIX and Linux systems, your user ID must have root authority to complete installation. Follow your local security guidelines to acquire root authority; either login as root, or log in as another user and become root.

Procedure

- On Windows, to uninstall IBM WebSphere MQ Telemetry you must modify the existing IBM WebSphere MQ. Deselect IBM WebSphere MQ Telemetry from the list of components that is installed in the main installation.
- On AIX and Linux, run the **rpm -qa | grep MQSeriesXR | xargs rpm -e** command to remove all IBM WebSphere MQ Telemetry components installed. Optionally, you can specify which IBM WebSphere MQ Telemetry component you want to remove, for example **rpm -e MQSeriesXRClients** or **MQSeriesXRService** to uninstall only the clients or the service respectively. For more information refer to the uninstall procedure of the main product, but replace step 4 with the commands in this step. "Uninstalling IBM WebSphere MQ on Linux" on page 307

Uninstalling WebSphere MQ Advanced Message Security

Information provided guides you through the uninstallation process of IBM WebSphere MQ Advanced Message Security component.

Related tasks:

"Uninstalling on AIX"
On AIX platforms, you can remove IBM WebSphere MQ Advanced Message Security component either using SMIT or the command line.

"Uninstalling on HP-UX" on page 316
Use the swremove command to remove IBM WebSphere MQ Advanced Message Security component on HP-UX platforms.

"Uninstalling on Linux" on page 317
Use the rpm command to remove IBM WebSphere MQ Advanced Message Security component on Linux platforms.

"Uninstalling on Windows" on page 317
You can uninstall IBM WebSphere MQ Advanced Message Security component using the GUI uninstallation wizard, or a command-line interface.

Uninstalling on AIX

On AIX platforms, you can remove IBM WebSphere MQ Advanced Message Security component either using SMIT or the command line.

Procedure

1. Stop all WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any IBM WebSphere MQ activity associated with the installation you are uninstalling:
   a. Log in as a user in the group mqm.
   b. Set up your environment to work with the installation you want to uninstall. Enter the following command:
      ```
      . MQ_INSTALLATION_PATH/bin/setmqenv
      ```
      where. MQ_INSTALLATION_PATH refers to the location where WebSphere MQ is installed.
   c. Display the state of all queue managers on the system. Enter the following command:
      ```
      dspmq -o installation
      ```
   d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:

```
endmqm QMgrName
```

e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:

```
endmqlsr -m QMgrName
```

3. Log in as root.
4. UninstallWebSphere MQ AMS component using either **installp** or **smit**. If WebSphere MQ AMS component was installed in a non-default location, you must use **installp** to uninstall.
   - Uninstall using **installp** by entering one of the following commands:
     – For an installation in the default location /usr/mqm

     ```
     installp -u mqm.ams.rte
     ```

     – For an installation in a non-default location:

     ```
     installp -R usil -u mqm.ams.rte
     ```

     where *usil* is the path of the User Specified Install Location (USIL) specified when the product was installed.
   - Uninstall using **smit**:
     a. Select the required **smit** window using the following sequence:

     ```
     Software Installation and Maintenance
       Software Maintenance and Utilities
         Remove Installed Software
     ```

     b. List the software in the **SOFTWARE name** field:
        1) Enter **.**
        2) Press **F4**

     c. Select the file sets to uninstall from the list (those beginning with mqm), and press **Enter**. There is an option at this stage to do a preview. Leave the option set to the default value of **Yes** to preview the file sets you are uninstalling, or select **No** to not preview these file sets.

     d. Press **Enter** on the **Remove Installed Software** panel, it asks whether you are sure, press **Enter**.

Results

The WebSphere MQ Advanced Message Security component has been uninstalled.

Uninstalling on HP-UX

Use the swremove command to remove IBM WebSphere MQ Advanced Message Security component on HP-UX platforms.

Procedure

1. Stop all WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any WebSphere MQ activity associated with the installation you are uninstalling:
   a. Log in as a user in the group mqm.

   b. Set up your environment to work with the installation you want to uninstall. Enter the following command:

   ```
   . MQ_INSTALLATION_PATH/bin/setmqenv
   ```

   where. MQ_INSTALLATION_PATH refers to the location where WebSphere MQ is installed.

   c. Display the state of all queue managers on the system. Enter the following command:

   ```
   dspmq -o installation
   ```

   d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:

   ```
   endmqm QMgrName
   ```

e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:

```
 endmqlsr -m QMgrName
```

3. Log on as root.
4. Run the following command:

```
swremove MQSERIES.MQM-AMS
```

Results

The WebSphere MQ Advanced Message Security component has been uninstalled.

Uninstalling on Linux

Use the rpm command to remove IBM WebSphere MQ Advanced Message Security component on Linux platforms.

Procedure

1. Stop all WebSphere MQ applications associated with the installation you are uninstalling.
2. For a server installation, end any WebSphere MQ activity associated with the installation you are uninstalling:
   a. Log in as a user in the group mqm.
   b. Set up your environment to work with the installation you want to uninstall. Enter the following command:

   ```
   . MQ_INSTALLATION_PATH/bin/setmqenv
   ```

   where. MQ_INSTALLATION_PATH refers to the location where WebSphere MQ is installed.

   c. Display the state of all queue managers on the system. Enter the following command:

   ```
   dspmq -o installation
   ```

   d. Stop all running queue managers associated with the installation you want to uninstall. Enter the following command for each queue manager:

   ```
   endmqm QMgrName
   ```

   e. Stop any listeners associated with the queue managers. Enter the following command for each queue manager:

   ```
    endmqlsr -m QMgrName
   ```

3. Log in as root.
4. Run the following command:

   ```
   rpm -e <package_name>
   ```

   where <package_name> is one of the following:

   - MQSeriesAMS-7.5.0-0.i386.rpm
   - MQSeriesAMS-7.5.0-0.x86_64.rpm
   - MQSeriesAMS-7.5.0-0.ppc.rpm
   - MQSeriesAMS-7.5.0-0.s390.rpm

Results

The WebSphere MQ Advanced Message Security component has been uninstalled.

Uninstalling on Windows

You can uninstall IBM WebSphere MQ Advanced Message Security component using the GUI uninstallation wizard, or a command-line interface.

Using the installation wizard:

Procedure

1. Insert the WebSphere MQ Server DVD into the DVD-ROM drive.
2. If autorun is enabled, the installation process starts.

   Otherwise, double-click the **Setup** icon in the root folder of the DVD to start the installation process.

   The WebSphere MQ Installation Launchpad window is displayed.
3. Click the **WebSphere MQ Installation**.
4. Click **Launch IBM WebSphere MQ Installer**. Click **Next** until the WebSphere MQ Program Maintenance panel is displayed with a welcome message.

   If this panel is not displayed, WebSphere MQ for Windows, Version 7.5 is not installed on this machine. When presented with the option, select to remove/maintain or upgrade.
5. Select **Maintain or upgrade an existing instance**, then click **Next**.
6. If there are any existing queue managers, the Removing Server feature panel is displayed.

   Click one of the following options, then click **Next**:

   - **Keep** - keep existing queue managers and their objects.
   - **Remove** - remove existing queue managers and their objects.

   The Program Maintenance panel is displayed, with a summary of the installation to be removed.
7. Click **Modify** and click **Next**.
8. On the list of available WebSphere MQ features, click Advanced Message Security, select **Do not install this feature (remove if already intalled)** , and click **Next**. The Ready to modify WebSphere MQ panel appears with the summary of your changes.
9. Click **Modify** and **Next** on the following panel to continue.

Results

Selected features of WebSphere MQ Advanced Message Security component have been removed.

Migrating and upgrading WebSphere MQ

To migrate a queue manager to run on a new level of code, you must first upgrade WebSphere MQ to install the new code level. When you have verified the upgrade is successful, migrate the queue manager and all the applications and resources associated with it. Before starting this process, create a migration plan, based on the information in this migration guide. If you are applying maintenance, no migration is necessary, but you must test applications with the new level of WebSphere MQ code.

The IBM WebSphere MQ Migration Guide provides information to help you plan the process of migrating from an older version to a new version of the product.

- For an introduction to the guide and its contents, see the developerWorks® blog article IBM WebSphere MQ Migration Guide.
- To view the guide in your web browser, click the following link: IBM WebSphere MQ Migration Guide - HTML version.
- To download the guide as a PDF file, click the following link: IBM WebSphere MQ Migration Guide - PDF file.

If you want to see what changes have occurred between two versions of IBM WebSphere MQ, see “List of changes by version, release, and maintenance level” on page 554. The changes that are listed affect the behavior of existing applications or the automation of management tasks. The lists do not include changes to IBM WebSphere MQ Explorer, or new features that do not change the way existing applications behave. Study the lists of changes to plan what migration tasks you must perform.

If you are not familiar with IBM WebSphere MQ migration, read the topics in “Introduction to WebSphere MQ migration” on page 320.

If you are familiar with IBM WebSphere MQ migration, start with a planning topic listed in Migration planning topics by version and platform. If you already have a migration plan, see “Migration tasks” on page 428 for step by step guides about how to migrate queue managers, IBM WebSphere MQ clients, and applications.

If you are running IBM WebSphere MQ Version 7.0.1 on Linux, UNIX, or Windows , and you are unfamiliar with multiple installations of IBM WebSphere MQ on the same server, examine these three migration scenarios:

1. “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394
2. “UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version” on page 401
3. “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409

Also, consider how multiple installations on the same server can assist you deploying fix packs; see “UNIX, Linux, and Windows: Staging maintenance fixes” on page 718. If you are installing a fix pack that includes optional new function, see “Migrating queue managers to new-function fix packs” on page 703.

“IBM WebSphere MQ maintenance tasks” on page 675 describes how to apply and remove maintenance by version and platform, for platforms other than z/OS.

“Migration commands, utilities, and reference information” on page 721 describe commands and functions that assist you with migration.

Migration planning topics by version and platform

IBM WebSphere MQ for z/OS and IBM WebSphere MQ for IBM i remain at Version 7.1. For information about migrating IBM WebSphere MQ to Version 7.1, and other versions, see "Where to find a topic about a specific migration path" on page 322. For information about migrating to Version 7.5 see Table 51.

Table 51. Planning topics by version and platform

Migrating from Version 6.0 to Version 7.5	Migrating from Version 7.0.1 to Version 7.5	Migrating from Version 7.1 to Version 7.5
AIX	AIX	AIX
HP-UX	HP-UX	HP-UX
Linux	Linux	Linux
Solaris	Solaris	Solaris
Windows	Windows	Windows

Introduction to WebSphere MQ migration

The introductory topics explain the concepts you must understand before planning migration tasks, where to find migration topics, and which migration paths are supported.

If you are running IBM WebSphere MQ Version 7.0.1 on Linux, UNIX, or Windows , and you are unfamiliar with multiple installations of IBM WebSphere MQ on the same server, examine these three migration scenarios:

1. "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
2. "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
3. "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409

Also, consider how multiple installations on the same server can assist you deploying fix packs; see "UNIX, Linux, and Windows: Staging maintenance fixes" on page 718. If you are installing a fix pack that includes optional new function, see "Migrating queue managers to new-function fix packs" on page 703.

Migration paths

Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

Use the links in Table 52 on page 322 and Table 53 on page 324 in "Where to find a topic about a specific migration path" on page 322, to find product documentation topics explaining how to migrate your system between different levels of WebSphere MQ. Each pair of release levels in the tables is called a migration path. If you migrate a system directly from one release to another, it is a direct migration path. If you migrate a system from one release to another, migrating first to an intermediate release, it is an indirect migration path.

Direct and indirect migration paths

The direct migration paths that are documented in the tables have been tested. The paths that are documented in this version of the product documentation are from one supported release to another. The paths are between releases that are currently within their support life, or only recently beyond their end-of-service dates. The paths between older releases are described in previous editions of the documentation. The links for those paths take you to previous editions of the documentation, where other direct migration paths were previously documented.

If you need to migrate a system from an old version WebSphere MQ, and no direct migration path is documented, you have two choices:

1. Migrate through one or more intermediate releases, using the published migration information.
2. Migrate directly from the old release to the new one.

If you choose the first option, you must start all the queue managers to migrate queue manager data to the intermediate release level. Depending on the release level, you might also need to start a channel on each queue manager to cause channel migration to take place.

If you choose the second option, you are choosing to follow an untested migration path.

Important:

1. If a queue manager is a member of a cluster, and is running at a release earlier than Version 6.0, you must migrate the queue manager to Version 6.0, Version 7.0, or Version 7.0.1, before migrating it to Version 7.5. You must start the queue manager after the first migration step, before proceeding to Version 7.5.
2. If you plan to migrate a queue manager on Linux for System x from 32 bit to 64 bit, you must do so by migrating from Version 6.0 to Version 7.0.1; see Linux: Migrating from a 32 bit to a 64 bit queue manager on System x.

In both cases, you must read the topics from a combination of direct migration paths to discover what other changes you might need to make.

Reverting to a previous version

Reverting to a previous release (backward migration) is generally more difficult than migrating forward. On platforms other than z/OS, you cannot revert a queue manager to a previous release that differs in the version or release. Instead you are usually advised to back up your existing queue managers before upgrading them. The forward migration tasks start with the step of backing up your existing queue managers. Should you revert to the previous release, you can restore the queue managers to their previous state from the backup copy. You cannot merge any messages or changes transacted on the new release to the system restored at the previous release level.

On z/OS, since V7.0.1, the process of migrating forward has been made more granular. You can restrict a queue manager in its use of new functions. Since V7.0.1, during migration, you must decide when to turn on new function in the later release. Until you turn new function on, using the **OPMODE** switch, you can revert a queue manager to its previous level. The previous level must be V6.0 or later. After you have turned on new function, you cannot revert a queue manager to its previous level. With this staged approach you can run production systems, for a while, at the new release level without exploiting the new function in the release. You have the assurance that if you encounter problems, you can revert the system to its earlier release level. When you are satisfied that existing applications run satisfactorily at the new release level, you can change the setting of the **OPMODE** switch. Then you can modify existing applications, or write new applications to use the new functions in the release.

Related concepts:

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

Where to find a topic about a specific migration path

You can find topics about direct migration to the current release of IBM WebSphere MQ in this release of the IBM WebSphere MQ product documentation. Only supported migration paths are documented. Paths between other releases are in previous versions of the product documentation. This topic has links to topics about all the direct migration paths.

Migration paths:IBM WebSphere MQ on platforms other than z/OS

The latest release of IBM WebSphere MQ for IBM i is Version 7.1. See earlier versions of the product documentation, which are online, for migration of IBM WebSphere MQ for IBM i to Version 7.1.

The links in Table 52 open product documentation topics. Some of the links go to topics in previous levels of the product documentation. No topics solely about migration were written for product documentation up to and including V5.3. Notice that backward migration is not possible for IBM WebSphere MQ on platforms other than z/OS.

If a migration path is marked as "not possible", you cannot restore a queue manager to the previous version from the current version. If a queue manager has not been started, you can uninstall the current version and reinstall a different version of IBM WebSphere MQ. To be precise, read the entries in Table 52 as information about whether you can start a queue manager on a different version of IBM WebSphere MQ than the one it was last started at. Do not read the entries as information about which version of IBM WebSphere MQ can be installed. It does not matter what versions of IBM WebSphere MQ are installed between when a queue manager was last started and when it is next started.

If a migration path between two different versions is marked with a dash, it means that direct migration between the versions is not supported by IBM WebSphere MQ. You must follow an indirect migration path, which involves migrating IBM WebSphere MQ more than once.

Table 52. Migration paths:IBM WebSphere MQ on platforms other than z/OS

From / To	6.0	7.0.0	7.0.1	7.1	7.5
5.3	WebSphere MQ v6.0 migration guide	Migrating from WebSphere MQ Version 5.3 to WebSphere MQ Version 7.0	-	-	-

Table 52. Migration paths:IBM WebSphere MQ on platforms other than z/OS (continued)

From / To	6.0	7.0.0	7.0.1	7.1	7.5
6.0	-	Migrating from WebSphere MQ Version 6.0 to WebSphere MQ Version 7.0	Planning for migration from WebSphere(r) MQ version 6.0 to WebSphere(r) MQ version 7.0.1 by platform	Planning for migration from WebSphere(r) MQ version 6.0 to WebSphere(r) MQ version 7.1 by platform	"Planning IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5 migration by platform" on page 418
7.0.0	Not possible	-	Planning for migration from WebSphere(r) MQ version 7.0 to WebSphere(r) MQ version 7.0.1 by platform	Direct migration from 7.0.0 to 7.1 is supported. Follow the instructions in Planning for migration from WebSphere(r) MQ version 7.0 to WebSphere(r) MQ version 7.1 by platform	Direct migration from 7.0.0 to 7.5 is supported. Follow the instructions in "Planning IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 migration by platform" on page 384
7.0.1	Not possible	Planning for migration from WebSphere(r) MQ version 7.0 to WebSphere(r) MQ version 7.0.1 by platform [1]	-	Planning for migration from WebSphere(r) MQ version 7.0 to WebSphere(r) MQ version 7.1 by platform	"Planning IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 migration by platform" on page 384
7.1	Not possible	Not possible	Not possible	-	"Planning IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5 migration by platform" on page 374
	Tip: You can revert a queue manager from Version 7.0.1 to Version 7.0. You must undo any application or administrative changes you made that took advantage of new functions in Version 7.0.1.				

Migration paths:IBM WebSphere MQ for z/OS

The latest release of IBM WebSphere MQ for z/OS is Version 7.1. See earlier versions of the product documentation for migration to Version 7.1.

The links in Table 53 on page 324 open product documentation topic that describe how to migrate IBM WebSphere MQ for z/OS between releases. All of the links go to topics in previous levels of the product documentation. A number of topics refer you to reverting to a previous release.

If a migration path between two different versions is marked with a dash, it means that direct migration between the versions is not supported by IBM WebSphere MQ. You must follow an indirect migration path, which involves migrating IBM WebSphere MQ more than once.

Table 53. Migration paths: IBM WebSphere MQ for z/OS

From / To	< 5.3.1	6.0	7.0.0	7.0.1	7.1
< 5.3.1	-	Migrating from an unsupported release of MQSeries®	Upgrading from an unsupported release of MQSeries		
5.3.0 / 5.3.1	Chapter 3. Migrating from a previous version	Migrating to Version 6			
6.0	Reverting to previous versions	-	Upgrading to WebSphere MQ for z/OS Version 7.0	z/OS(r): Planning for migration from WebSphere(r) MQ version 6.0 to WebSphere(r) MQ version 7.0.1	z/OS(r): Planning for migration from WebSphere(r) MQ version 6.0 to WebSphere(r) MQ version 7.1
7.0.0		Reverting to previous versions	-	z/OS(r): Planning for migration from WebSphere(r) MQ version 7.0 to WebSphere(r) MQ version 7.0.1	-
7.0.1		z/OS(r): Restoring a version 7.0.1 queue manager to version 6.0	z/OS(r): Restoring a version 7.0.1 queue manager to version 7.0	-	z/OS(r): Planning for migration from WebSphere(r) MQ version 7.0.1 to WebSphere(r) MQ version 7.1
7.1		z/OS(r): Reverting a version 7.1 queue manager to version 6.0	-	z/OS(r): Reverting a version 7.1 queue manager to version 7.0.1	-

The version naming scheme for WebSphere MQ on platforms other than z/OS

On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

The full version of WebSphere MQ on platforms other than z/OS is described by a four-digit VRMF code. The VRMF acronym stands for:

```
Version.Release.Modification.Fix
```

`5.3`, `7.0.0.1`, and `7.0.1.0` are examples of full WebSphere MQ version codes on platforms other than z/OS.

You can find the full version level of a WebSphere MQ installation by typing the command **DSPMQVER**, or **DSPMQMVER** on IBM i. It returns the full four-digit VRMF code.

Versions and releases of WebSphere MQ are known by the first two digits of the VRMF code. The two digits are sometimes prefixed by a `V`, such as `V5.3`. A version of WebSphere MQ always has a release level, even if it is the first release in a version.

The first release is normally labeled `Vx.0`, for example, WebSphere MQ V7.0. Occasionally, the first release of a version on a specific platform is not labeled `Vx.0`. It is numbered to correspond the command level that has been implemented on the platform.

In documentation, the release level is sometimes dropped from the VRMF code, for example, `V5`. Dropping the release level can lead to ambiguity, if the context is not clear. For example, V5 might mean the whole of V5, or the release level V5.0, as opposed to the release level V5.2, or V5.3.

The third digit in the VRMF identifies the modification level of a release. A change in the third digit does not change the release. After upgrading WebSphere MQ to modification level `7.0.1`, the release of WebSphere MQ remains 7.0. However the command level does change to `7.0.1`.

The significance of the distinction between release and modification level concerns migration, and the service life of a product. Queue manager objects, such as queue managers, channels, queues, and messages do not require migration to upgrade to a new modification level. Nor do they require migration if the modification level is removed[1]. Migration might be required for a version or release level change.

Reverse migration of a version/release change is not possible on platforms other than z/OS. To be able to restore an earlier version or release level of a queue manager, you must back it up before upgrading. If you do restore it, you restore the queue manager, and its data, to the state it was in when you backed it up.

The other difference concerns service life. A new version or release has a new service end date. New modification levels generally do not result in a new service end date. But if a modification level is announced, then a new service end date might be announced too.

The fourth digit in the VRMF code is the fix level. Fix levels do not affect the command level of the queue manager. No migration is required, and fix levels do not affect the service end date of a release.

Trailing zeros in the VRMF code are never significant, but are sometimes quoted for clarity. For example, you might see `7.0.0` to distinguish it from `7.0.1`, and `7.0.1.0` to distinguish it from `7.0.1.1`. `7.0.0` is no different to `7.0` or `7.0.0.0`, and `7.0.1` and `7.0.1.0` are the same level.

Modification levels and fix levels are known by three and four-digit VRMF codes. `7.0.1` is a modification level and `7.0.1.2` is a fix level. Modification levels are shipped as refresh packs, and fix levels as fix packs.

A refresh or fix pack is named using a two part name that uniquely identifies it. The first part of the name is a truncated VRMF. The second part of the name is the name new refresh or fix pack. So, for example, the name of the fix pack `7.0.1.2` for Windows is `7.0.1-WS-MQ-Windows-FP0002`, and the name of the refresh pack `7.0.1` for Windows is `7.0-WS-MQ-Windows-RP0001`.

Refresh packs and fix packs for a particular version/release are cumulative, from the initial release. You can apply any higher numbered refresh, or fix pack, of the same version/release to upgrade directly to that version level. You do not have to apply the intervening fixes. Refresh packs and fix packs are obtained as service through Fix Central.

The latest modification level is also used to refresh the version of WebSphere MQ available through Passport Advantage, or on physical media. When you order WebSphere MQ you receive the latest

[1] Applications using new functions introduced in a modification level do not work on an earlier level.

modification level. The result of installing the manufacturing refresh is almost the same as applying the refresh pack to an earlier fix level of WebSphere MQ. There is one important difference. Refresh packs are applied using a maintenance procedure, manufacturing refreshes are installed using an installation procedure. You can "unapply" a refresh pack to return to the previous fix level you had installed. You can only uninstall a manufacturing refresh, which removes WebSphere MQ from your system.

In addition to fixes packaged as refresh packs and fix packs, you can also obtain interim fixes for WebSphere MQ from Fix Central. Interim fixes are also known as emergency or test fixes, and are known collectively as interim fixes. The naming scheme for refresh and fix packs extends to interim fixes. Interim fixes are known either by their fix name, or by the list of APARs they fix. The interim fix for APAR IC60646 is called `7.0.0.2-WS-MQ-Windows-LAIC60646`, telling you it is based on `7.0.0.2`. Interim fixes are like PTFs on z/OS in the respect that they do not change the VRMF code.

When you apply new fix packs or refresh packs, all interim fixes are removed. The documentation with the fix pack or refresh pack tells you if the APARS associated with the interim fixes you have applied have been fixed. If they have not, check to see if there are new interim fixes, at the new level, for the APARs that concern you. If there are not, consult service. They might either tell you to reapply the interim fix, or supply a new interim fix.

Related concepts:

"Migration paths" on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

Internet Protocol Version 6 (IPv6) migration

This section deals with using IPv4 and IPv6 when you are thinking of installing IBM WebSphere MQ

General Introduction

The Internet Protocol Version 6 (IPv6) is designed by the Internet Engineering Task Force (IETF) to replace the current version Internet Protocol, Version 4 (IPv4). IPv4 has been around for over 20 years and is one of the primary methods for machines to communicate to each other over the internet. IPv4 is limited to 32-bit addressing for internet addresses. These addresses are needed by all new machines added to the internet and they are beginning to run out. The IETF is the controlling standards body for the Internet and to meet the growing demand for internet addresses has increased the number of digits used for Internet addresses from 32 to 128 bits. IPv6 offers a far larger number (2^{128}) of internet addresses and should solve the address shortage for the foreseeable future. IPv6 is expected to gradually replace IPv4, with the two protocols coexisting for a number of years while this transition period exists. IPv6 also simplifies header formats and improves support for extensions and options, flow labeling capability, and consolidated authentication and privacy capabilities

IBM WebSphere MQ has the ability for queue managers to communicate using the IPv6 protocol in addition to the existing, IPv4, protocol.

Further information on IPv6 can be found at IPv6 and IPv6 Forum.

IBM WebSphere MQ platforms that support IPv6

This section lists the IBM WebSphere MQ platforms that support IPv6.

IPv6 is supported on the following IBM WebSphere MQ platforms:

- IBM WebSphere MQ for AIX
- IBM WebSphere MQ for Linux
- IBM WebSphere MQ for Sun Solaris
- IBM WebSphere MQ for HP-UX
- IBM WebSphere MQ for Windows
- IBM WebSphere MQ for IBM i
- IBM WebSphere MQ for z/OS

Key points in migrating to IPv6 and using IBM WebSphere MQ

This section lists some key points to be aware of when you are thinking of installing IBM WebSphere MQ and using IPv6.

- IBM WebSphere MQ recognizes IPv6 hexadecimal addresses (for example fe80:43e4:0204:acff:fe97:2c34:fde0:3485) as well as IPv4 dotted decimal addresses (for example 9.20.9.30).
- For a system running both IPv4 and IPv6 system, the connection name (CONNAME) you specify for a given channel determines the IP protocol for the channel making a connection.

Considerations when implementing IPv6 in a network

This section lists some things that you should consider when you are thinking of installing IBM WebSphere MQ on an IPv6 network.

- To ensure consistency across the network, you should plan the introduction of IPv6 for the whole network, especially where clusters are involved. For example, although a queue manager is now IPv6 capable, this doesn't imply that the queue managers it can communicate with are also IPv6 capable.
- When setting the domain name server (DNS) or equivalent, consider whether the system on which the target queue manager is running can resolve to an IPv4 address, an IPv6 address or a dual IPv4 and IPv6 address.
- If the system that you are installing IBM WebSphere MQ on does not support IPv6, IBM WebSphere MQ will only be able to connect using IPv4.
- For a queue manager running on an IPv6 enabled system to be able to communicate with a queue manager running on an IPv4 enabled system, the IPv4 enabled system must have a hostname that resolves to an IPv4 address only.
- If there are multiple domain name servers in a IBM WebSphere MQ network, each hostname used in a channel definition must resolve to the same address (or addresses), regardless of which DNS is used.

Migrating a queue manager to IPv6

This section deals with migrating a queue manager when you are thinking of installing IBM WebSphere MQ on an IPv6 network.

The IPv6 protocol can only be utilized by IBM WebSphere MQ or later. In order to make use of the IPv6 protocol, IBM WebSphere MQ must be installed on a system that is IPv6 capable.

The preferred IP version that two systems use for communicating (if both IPv4 and IPv6 are available) is determined by a new queue manager attribute IPADDRV. This parameter only has an effect if the hostname resolves ambiguously to both an IPv4 address and an IPv6 address.

To migrate a queue manager to use the IPv6 protocol:

1. Configure dual IPv4 and IPv6 protocols on the system where the queue manager to be migrated resides.
2. Install IBM WebSphere MQ.

3. Add an entry to the DNS to resolve the hostname of the system that is to be migrated, to both an IPv4 address and an IPv6 address.
4. Set the IPADDRV parameter to IPv6 (or set the LOCLADDR parameter to resolve to an IPv6 address).

   CAUTION:
   Not all IPv6 software can interpret an IPv4 mapped IPv6 address. If the combination of CONNAME and LOCLADDR results in an IPv4 mapped IPv6 address, ensure that the system hosting the target queue manager is capable of handling this.

   Using mapped addresses can require protocol translators in the IP network.

Migration scenarios (non-cluster topology)

It is possible to come up with a number of different interconnection possibilities, and the following sections aim to help you understand how IBM WebSphere MQ will work in each case.

Non-cluster migration scenario 1

Three systems exist that are IPv4 only capable. Each system hosts a queue manager (QM1, QM2, and QM3) and each queue manager connects to the other two. All CONNAMEs in the cluster channel definitions are made using DNS names rather than IP addresses.

Enable QM1 to be able to use channels running over IPv6 as follows

1. Upgrade the host system to have dual IPv4 and IPv6 stacks.

   Important: A listener is required for each IP stack.
2. Install the latest version of IBM WebSphere MQ.
3. Update the DNS table so that it has two entries for the system running QM1; one entry for its IPv4 address and one for its IPv6 address. This enables a DNS name request to return both IPv4 and IPv6 addresses for this host.
4. Set the queue manager IPADDRV attribute to IPv6.

Note: Even with these changes made to support IPv6 addressing, QM1 will still be able to communicate with queue managers (both existing and new ones) that are only IPv4 capable.

Enable QM2 to be able to use channels running over IPv6 as for QM1 above.

- Communications between QM1 and QM2 will now be over IPv6.
- Communications between QM1 and QM3 will still be over IPv4.
- Communications between QM2 and QM3 will still be over IPv4.

With the queue manager IPADDRV attribute set to IPv6, the preference has been set for the queue manager to connect using the IPv6 protocol. If a channel from QM1 to QM3 has LOCLADDR set to a host name which resolves to an IPv6 address, or both IPv4 and IPv6 addresses (with the IPADDRV attribute set to IPv6, the IPv6 address will be returned as that is the preference), this channel will attempt to use the IPv6 protocol. If the IPv6 protocol installed on the QM1 host system is capable of using a mapped address then QM1 will communicate with QM3 over IPv6. Otherwise, the channel will fail to resolve CONNAME.

While QM3 remains a queue manager on an earlier version of the product, you will need to check that all CONNAMEs used to start a channel to QM3 do not resolve to an IPv6 address or dual IPv4 and IPv6 addresses where the IPv6 address could be returned. This would cause QM1 to attempt to start the channel over IPv6 which would fail, as it would be unable to resolve the CONNAME.

It is possible to upgrade a system to have dual IPv4 and IPv6 capability and still run a queue manager on an earlier version of the product, on the system. While it is not recommended to run this type of configuration, as long as the addresses that are returned to this level of queue manager are either IPv4 or an IPv4 mapped version of an IPv6 address, this should work.

Non-cluster migration scenario 2

Three systems exist that are IPv4 only capable. Each system hosts a queue manager (QM1, QM2, and QM3) and each queue manager connects to the other two. All CONNAMEs in the cluster channel definitions are made using IP addresses.

Because addresses have been specified instead of DNS names, to allow a queue manager to connect to another using the IPv6 protocol you will need to duplicate the definitions that use IPv4 addresses between them and provide them with IPv6 addresses instead. The original definitions that use IPv4 addresses will continue to work, but if you intend to take advantage of the IPv6 protocol, you will need to connect using the new definitions.

Enable QM1 to be able to use channels running over IPv6 as follows

1. Upgrade the host system to have dual IPv4 and IPv6 stacks.

   Important: A listener is required for each IP stack.
2. Install IBM WebSphere MQ.
3. Duplicate the channel, transmission queue and, where applicable, any process definitions using IPv6 addresses where required.

Note: Even with these changes made to support IPv6 addressing, QM1 will still be able to communicate with existing queue managers that are only IPv4 capable.

Enable QM2 to be able to use channels running over IPv6 as for QM1 above.

1. Upgrade the host system to have dual IPv4 and IPv6 stacks.

   Important: A listener is required for each IP stack.
2. Install IBM WebSphere MQ.
3. Where necessary amend applications to write to the new remote queue (created above for QM1 with the IPv6 addresses).
4. Verify the channels can be started.

The queue managers can now connect as follows:

- QM1 can now connect with QM2 over either IPv4 or IPv6 depending on the channel the application writes its messages to.
- QM1 still connects with QM3 over IPv4 using the original definitions.

Migrating a cluster to IPv6

This section deals with migrating clusters when you are thinking of installing IBM WebSphere MQ on an IPv6 capable network.

The following gives an overview of approaches that can be taken when migrating a cluster to the latest version of IBM WebSphere MQ. Due to the variations that can occur within a cluster, the detail is deliberately general and should only be seen as a guide to the likely course of action you will need to take.

Migration scenarios (cluster topology)

Where an IPv6 capable system is to be added to a IBM WebSphere MQ cluster, all full repository systems in that cluster must be IPv6 capable.

The following scenarios are seen as the ones most likely to occur in customer installations. They describe the changes that are likely to be required.

Scenario 1

A cluster from an earlier version of the product is installed on IPv4 only capable, systems and you need to connect an IPv6 only capable system into the cluster. All CONNAMEs in cluster channel definitions are made using DNS names rather than IP addresses.

When adding a new IPv6 only system to the cluster, identify those queue managers that your new system will communicate with. These include:

- The queue managers your new system will send messages to.
- The queue managers your new system will receive messages from.
- The full repository queue managers

The systems that you have identified must be upgraded before introducing the new system.

Recommended migration procedure:

- Upgrade each of the systems hosting a full repository queue manager as shown in "Migrating a queue manager to IPv6" non-cluster scenario 1.
- Upgrade the remaining cluster systems which need to be IPv6 capable as shown in "Migrating a queue manager to IPv6" non-cluster scenario 1.

With this configuration:

- The new IPv6 only capable system will communicate with the cluster using IPv6 addressing
- All other IPv4 systems that connect into the cluster will continue to communicate using IPv4 addressing
- The systems in the cluster will be able to connect to each other using either IPv4 or IPv6 addressing. The decision as to which address is used depends on whether you have set IPADDRV to specify IPv4 or IPv6 connections.

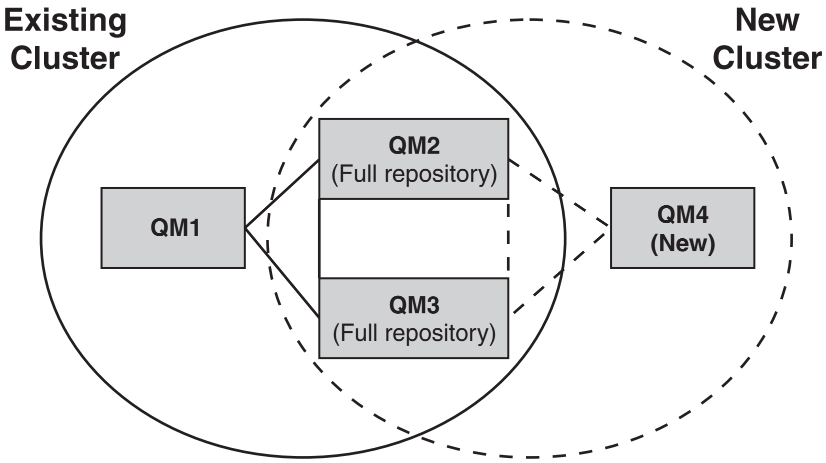
Scenario 2

A cluster from an earlier version of the product is installed on IPv4 only capable systems and you need to connect an IPv6 only capable system into the cluster. Your network does not support adding both IPv6 and IPv4 addresses using the same hostname or you are using IP addresses rather than DNS names in the cluster channel CONNAMEs.

The problem here is likely to be that all of the systems cannot be switched to IPv6 simultaneously and some at least must remain only IPv4 capable. The systems that your new IPv6 only system communicates with must be IPv4 and IPv6 capable. We do not recommend simply adding a new set of IPv6 channels into the cluster for the IPv6 system to use, as the IPv4 system would also try to use them, resulting in communication errors.

The recommended approach is:

- Define a new cluster which contains the IPv6 only capable system or systems with new IPv6 addresses and channel definitions. The existing cluster remains, and contains the IPv4 only system definitions. The image below gives a pictorial representation of this. QM1, QM2, and QM3 represent the original IPv4 cluster. QM2, QM3, and QM4 represent the new cluster created to allow the IPv6 only capable system (QM4) to connect into your configuration.
- If you are using DNS names, you can give each of the systems separate DNS names for IPv4 and IPv6 (for example system1_IPv4.ibm.com and system1_IPv6.ibm.com).
- Define a new CLUSRCVR channel and any corresponding CLUSSDR channels using the new IPv6 names or IP addresses on each system in the new cluster. In this way the systems with only IPv4 or IPv6 capability do not see channels which they are not able to use and no communications error will result.



Note: There are both IPv4 and IPv6 definitions connecting the full repositories so that definitions for both new and existing cluster definitions are replicated between them. Also be aware that the queue managers QM1 and QM4 cannot communicate directly because they do not share a common network. They could communicate indirectly, for example by using ALIAS queues defined in the queue managers QM2 and QM3. In the configuration shown above you would need to pay attention to the ordering of application messages flowing between QM2 and QM3 because multiple routes exist, if this is relevant you could use BIND_OPEN to fix the route.

Abbreviated migration scenarios

This section gives some abbreviated scenarios for when you are thinking of installing clusters on IBM WebSphere MQ

Abbreviated scenarios: Effects of CONNAME and LOCLADDR settings

The following table provides an overview of what will occur for the different TCP/IP stacks (IPv4 only, IPv6 only and dual IPv4 and IPv6 stacks) and given the settings for CONNAME and LOCLADDR the expected connection result.

Note: Using mapped addresses can require protocol translators in the IP network.

Table 54. Effects of CONNAME and LOCLADDR settings. Table showing the connection results for different TCP/IP stacks given the settings of CONNAME and LOCLADDR

Stack Type	CONNAME setting	LOCLADDR setting	Connection result
IPv4 only stack	IPv4 address		Channel binds to IPv4 stack
	IPv6 address		Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses		Channel binds to IPv4 stack
	IPv4 address	IPv4 address	Channel binds to IPv4 stack
	IPv6 address	IPv4 address	Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses	IPv4 address	Channel binds to IPv4 stack
	Any address	IPv6 address	Channel fails to resolve LOCLADDR
	IPv4 address	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv4 stack

Table 54. Effects of CONNAME and LOCLADDR settings (continued). Table showing the connection results for different TCP/IP stacks given the settings of CONNAME and LOCLADDR

Stack Type	CONNAME setting	LOCLADDR setting	Connection result
	IPv6 address	Host name resolves to both IPv4 and IPv6 addresses	Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv4 stack
Dual IPv4 and IPv6 stack	IPv4 address		Channel binds to IPv4 stack
	IPv6 address		Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses		Channel binds to stack determined by IPADDRV
	IPv4 address	IPv4 address	Channel binds to IPv4 stack
	IPv6 address	IPv4 address	Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses	IPv4 address	Channel binds to IPv4 stack
	IPv4 address	IPv6 address	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	IPv6 address	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	IPv6 address	Channel binds to IPv6 stack
	IPv4 address	Host name resolves to both IPv4 and IPv6 addresses	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack
IPv6 only stack	IPv4 address		Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address		Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses		Channel binds to IPv6 stack
	Any address	IPv4 address	Channel fails to resolve LOCLADDR

Table 54. Effects of CONNAME and LOCLADDR settings (continued). Table showing the connection results for different TCP/IP stacks given the settings of CONNAME and LOCLADDR

Stack Type	CONNAME setting	LOCLADDR setting	Connection result
	IPv4 address	IPv6 address	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	IPv6 address	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	IPv6 address	Channel binds to IPv6 stack
	IPv4 address	Host name resolves to both IPv4 and IPv6 addresses	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack

Abbreviated scenarios: System configurations

Table 56 on page 334 gives a number of abbreviated scenarios based on the configuration of the installed queue managers and the IP configuration they are running on. The list is not intended to be exhaustive, but to give a number of examples of what to expect based on the configurations shown.

The abbreviations are combined in Table 56 on page 334 to give the configuration of the systems involved in trying to establish communication. For example:

- v53 + IPv6: Represents a queue manager from an earlier version of the product on a system with a TCP/IP version 6 stack
- v6 + Dual: Represents a queue manager from the latest version of the product on system with a dual TCP/IP version 4 and version 6 stack

Table 55. Abbreviations used in system configurations. Table showing the abbreviation used in the documentation together with the meaning of the abbreviation.

Abbreviation	Meaning
v53	queue manager from an earlier version of the product
v6	queue manager from the latest version of the product
IPv4	a system using an IPv4 only stack
IPv6	a system using an IPv6 only stack
Dual	a system using both an IPv4 and an IPv6 stack
IPv4DNS	DNS returns an IPv4 address only for hostname of system holding the responding queue manager
IPv6DNS	DNS returns an IPv6 address only for hostname of system holding the responding queue manager

Table 55. Abbreviations used in system configurations (continued). Table showing the abbreviation used in the documentation together with the meaning of the abbreviation.

Abbreviation	Meaning
DualDNS	DNS returns an IPv4 and IPv6 address for hostname of system holding the responding queue manager
LOCLADDR4	The LOCLADDR parameter is set to IPv4 addressing
LOCLADDR6	The LOCLADDR parameter is set to IPv6 addressing
IPADDR4	IPADDRV is set to IPv4 addressing
IPADDR6	IPADDRV is set to IPv6 addressing

Table 56. System configurations. Table showing the results for different settings on the originating and responding queue manager

Originating queue manager		Responding queue manager			Result
Queue manager and Stack	LOCLADDR	IPADDRV	Queue Manager and Stack	DNS Return	
v53 + IPv6	Any	Not applicable			IP Error
v53 + IPv4 or v53 + Dual	Both LOCLADDR4 & LOCLADDR6	Not applicable	v53 + IPv4 or v53 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v53 + IPv4 or v53 + Dual	Blank or LOCLADDR4	Not applicable	v53 + IPv4 or v53 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v53 + IPv4 or v53 + Dual	Blank or LOCLADDR4	Not applicable	v53 + Dual	IPv6DNS	Unable to resolve CONNAME
v53 + IPv4 or v53 + Dual	Blank or LOCLADDR4	Not applicable	v53 + Dual or v6 + Dual v6 + IPv4	IPv4DNS or DualDNS	IPv4 connection can be established
v53 + IPv4 or v53 + Dual	LOCLADDR6	Not applicable			IP Error
v53 + IPv4 or v53 + Dual	Blank or LOCLADDR4 or both LOCLADDR4 & LOCLADDR6	Not applicable	v6 + IPv6	IPv6DNS	Unable to resolve CONNAME
v6 + IPv4	Blank or LOCLADDR4	Not specified	v53 + IPv4 or v53 + Dual or v6 + IPv4	IPv4DNS or DualDNS	IPv4 connection can be established
v6 + IPv4	LOCADD6	Not specified			Unable to resolve LOCLADDR
v6 + IPv4	Blank or LOCLADDR4	Not specified	v6 + IPv6	IPv6DNS	Unable to resolve CONNAME
v6 + IPv6	Blank or LOCLADDR6	Not specified	v53 + Dual	DualDNS	Attempts to start IPv6 channel and fails as there will be no IPv6 listener available

Table 56. System configurations (continued). Table showing the results for different settings on the originating and responding queue manager

Originating queue manager		Responding queue manager			Result
Queue manager and Stack	**LOCLADDR**	**IPADDRV**	**Queue Manager and Stack**	**DNS Return**	
v6 + IPv6	Blank or LOCLADDR6	Not specified	v53 + IPv4	IPv4DNS	Attempts to start IPv6 channel and fails as there will be no IPv6 listener available
v6 + IPv6 or v6 + Dual	LOCLADDR6	Blank or IPADDR6	v6 + IPv6 or v6 + Dual	IPv6DNS or DualDNS	IPv6 connection can be established
v6 + Dual	LOCLADDR6	IPADDR4	v6 + Dual	IPv4DNS or DualDNS	IPv6 connection can be established where mapped addressing can be used
v6 + Dual	Blank or LOCLADDR4	IPADDR4	v53 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v6 + Dual	Both LOCLADDR4 & LOCLADDR6	Blank or IPADDR4	v53 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v6 + Dual	LOCLADDR4	IPADDR4			Unable to resolve LOCLADDR
v6 + Dual	LOCLADDR6 or both LOCLADDR4 & LOCLADDR6	Blank or IPADDR6	v6 + IPv6 or v6 + Dual	IPv6DNS or DualDNS	IPv6 connection can be established

Maintenance, upgrade, and migration

Maintenance is a reversible change to the code level of WebSphere MQ. Maintenance requires no migration. Upgrading is the process of changing the code level of WebSphere MQ. Upgrades can be backed out, as long as no migration has taken place. The process of removing an upgrade varies by platform and how the upgrade was applied. Upgrades that change the command level of WebSphere MQ require queue manager migration before applications can reconnect. Migration is the process of updating queue managers, and other objects, such as applications or administrative procedures. Typically, migration cannot be reversed.

Maintenance is the application of a fix pack, interim fix or PTF. It has one main characteristic. Those fixes, whether applied using a maintenance installation tool, or installed using a manufacturing refresh on top of an installation, are at the same command level as the existing code. No migration is required after applying maintenance. The installation can be restored to its previous level and any changed queue managers or applications will continue to work at the restored code level.

Upgrading and migration are related but distinct. Upgrading is the process of taking an existing WebSphere MQ installation and upgrading to a new level of code. Unless you are upgrading the fix level of WebSphere MQ, but not its command level, an upgrade must be followed by migration. Migration is the process of converting queue managers, applications, and other objects to run at a new command level.

An upgrade can take four different forms:

1. Application of a fix pack, interim fix, or a program temporary fix (PTF) using maintenance installation tool. Upgrades applied this way might not be called upgrades, but just fixes. Fixes, applied using a maintenance installation tool, can be rolled back completely as long as no queue manager migration has taken place, and WebSphere MQ is returned to its previous code level.
2. Installation of new code on top of existing code. You might be able to roll back an upgrade applied in this way; it depends on the platform. Generally speaking, you cannot roll back the installation of new code. To restore the old code level, you must retain the old installation media, and any fixes you applied.
3. Removal of the old level of code, followed by installation of the new level. The installers on very few platforms require you to remove an old installation first. Needless to say, to restore the old code level, you must reinstall it and any fixes.
4. Side by side installation. On UNIX, Linux, and Windows you can install different code levels are stored alongside each other on the same server. On UNIX, Linux, and Windows, you associate a queue manager with an installation, and start the queue manager. In WebSphere MQ, running multiple queue managers at different command levels on the same server is termed queue manager coexistence. You must not infer from this you can select different installations to run a queue manager at different times. Once a queue manager has been run, it is subject to the rules regarding reverting to earlier or later command levels.

Migration always follows an upgrade that changes the queue manager command level, both automatic and manual changes. Migration is the transformation of queue manager data, applications, and the environment that the queue manager runs in. Migration, and maintenance, and upgrading are described in the following topics.

WebSphere MQ maintenance

Maintenance is the application of a reversible fix. The application and identification of maintenance fixes differs on z/OS from other platforms. Any changes to queue manager data are compatible with the previous code level.

Maintenance is the process of applying program temporary fixes on z/OS, or interim fixes or fix-packs, on other platforms. The way maintenance is identified using the V.R.M.F version naming scheme on z/OS, and on other platforms, is different; see the Related concepts. On z/OS, a maintenance change is a fourth digit change. On other platforms, it is either a third or fourth digit change.

An important characteristic of applying maintenance is that it must be reversible. Reversibility implies two things:

1. The previous level of code is fully restored.
2. Changes that are made to WebSphere MQ objects are compatible. Changes are things like the creation or deletion of persistent messages, changes to queue managers, channels, topics, and queues. New and modified objects continue to work correctly with the restored level of code.

The reversibility of a maintenance package limits the extent of functional changes that are included in a maintenance package. No irreversible changes are included in a maintenance package. But, reversibility has limits. A maintenance package might include new programming and administrative interfaces. If you build new or modified applications to use the new interfaces, those applications do not work, if the maintenance package is removed.

Multi-instance queue managers are a good example. Should you remove the version 7.0.1 fix pack that upgraded version 7.0, then multi-instance queue manager functions no longer work. However, the queue managers continue to function correctly as single instance queue managers in version 7.0.

On a smaller scale, a fix pack or interim fix might introduce a new configuration parameter to solve a problem. If you remove the fix pack or interim fix, although the new interface introduced by the change is not available any more, WebSphere MQ works with any objects that have been changed by the configuration parameter. For example, a new Java system property might introduce a parameter to set a code page for queue manager data conversion. The fix does not change any existing persistent queue manager state information. It can be removed, and the queue manager continues to work as before, but without the capability introduced in the fix.

Related concepts:

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

IBM WebSphere MQ upgrade

The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Reversibility of a upgrade has two parts; backout of the installation to the previous code level, and reversion of any queue managers that have been started at the new code level, to work with the previous code level again.

Backout of an installation is not required on UNIX, Linux, and Windows. You can continue to run queue managers with the previous code level, after installing a new code level. The flexibility is due to those platforms supporting multi-installations.

On other platforms, which do not support multiple installations, it depends on how the upgrade has been applied. If an upgrade is applied as a fix pack, which is only available for maintenance and fix level changes, it can be reversed. If the upgrade is installed as a manufacturing refresh, it must be uninstalled. You require installation materials, and any maintenance levels that you applied, to restore IBM WebSphere MQ to its previous level.

The rules regarding the reversibility of an queue manager to run on a previous code level is dependent on the platform.

On UNIX, Linux, and Windows, changes in version or release level are not fully reversible, but changes in maintenance or fix level are reversible under certain conditions. If the maintenance level upgrade is applied as a normal fix pack, removing the fix pack restores the previous level. If the maintenance or fix level change is applied as a manufacturing refresh, the refresh must be removed. You can then reinstall the previous level, together with any fixes, to restore the previous level. Once the previous level of code has been restored, queue managers can be restarted, and continue with processing messages from when they were last stopped. If the maintenance level upgrade, no matter if it is a fix pack or a manufacturing refresh, contains new function, it introduces a new command level. A maintenance level upgrade is applied to a specific installation. If any queue manager associated with the installation enables the new command level, the maintenance level upgrade cannot be removed from the installation. Maintenance level upgrades applied later than the one including the new function can be removed.

An irreversible upgrade implies that you must back up the queue managers, or your system, to be able to restore your queue managers. Taking a backup of a queue manager requires you to stop the queue manager. If you do not take a backup, you are not able to restore IBM WebSphere MQ to its previous level. Any changes you make on the new level cannot be restored onto the backup system. Changes include the creation or deletion of persistent messages, and changes to queue managers, channels, topics, and queues.

Note: The term upgrade does not imply that a IBM WebSphere MQ installation can be directly upgraded from one level to another. On some platforms, an upgrade requires that you remove the previous IBM WebSphere MQ installation. You can retain any queue managers that you have created.

Related concepts:

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"New function in maintenance level upgrades" on page 341
On platforms other than z/OS, IBM might introduce new functions between releases in maintenance level upgrades such as fix packs. A maintenance level upgrade including new function increases the maximum command level of an installation. When you apply the maintenance level upgrade, the installation supports the new command level. A queue manager associated with the installation is not able to use the new function until you have run **strmqm** to set the queue manager to the new command level.

IBM WebSphere MQ migration

Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

Whenever you upgrade WebSphere MQ to a new release that changes its command level, migration is performed by the queue manager. Whenever you upgrade WebSphere MQ to a new maintenance or fix level, which introduces a new function using a new command level, you can migrate the queue manager to use the new command level and thereby the new function.

You must read "List of changes by version, release, and maintenance level" on page 554 before upgrading your WebSphere MQ installation or migrating your queue managers, to identify what migration tasks you must plan for.

Using the model in Figure 63 on page 341 you can distinguish different migration questions, which are discussed in the following topics:

Operating environment migration
Upgrading the operating environment, or components in the environment such as installing a new level of JRE; see "WebSphere MQ operating environment migration" on page 344

Queue manager migration
Migrating a queue manager following an upgrade of the IBM WebSphere MQ installation to a new command level; see "Queue manager migration" on page 357.

IBM WebSphere MQ MQI client migration
Migrating a client configuration following installation of a new version or release of the WebSphere MQ MQI Client; see "WebSphere MQ MQI client migration" on page 359.

Application migration
Relinking, recompiling, or recoding a WebSphere MQ server or client application; see "Application migration and interoperation" on page 360. Application migration also includes migrating any API or channel exits

In addition, you must consider the impact of migrating one queue manager, or WebSphere MQ MQI client, on other clients or queue managers:

Compatibility, coexistence, and interoperability

See "Coexistence, compatibility, and interoperability" on page 345 for information about the compatibility of WebSphere MQ applications connected to queue managers and WebSphere MQ MQI clients on different command levels. The section also explains the concept of queue manager coexistence, and the interoperability of WebSphere MQ JMS applications with WebSphere Application Server.

Queue manager clusters

Can a queue manager cluster contain queue managers at different command levels? See "Queue manager cluster migration" on page 362 to answer this question, and how to migrate a cluster of queue managers.

Queue-sharing groups

Queue sharing groups involve multiple queue managers running on z/OS. How do you migrate queue managers that are part of a queue-sharing group to a new command level; see "Queue-sharing group migration" on page 363.

High-availability clusters

How do you migrate queue managers that are part of a high-availability cluster to a new command level, and maintain continuous and reliable service? See "Migrate a queue manager in a high availability configuration" on page 365, which covers both migration of multi-instance queue managers, and the migration of queue managers operating in high-availability clusters.

The remaining migration topics describe migration from other products or WebSphere MQ features, to being part of a queue manager.

WebSphere MQ publish/Subscribe broker

The version 6 publish/subscribe broker was separate from the queue manager. It used command messages to create and control publications and subscriptions. In migrating it to version 7, and integrating it with the queue manager, two major changes were introduced. The configuration and administration of publish/subscribe changed, and a new programming interfaced was introduced, integrated with the MQI. The first change requires any installation that used the version 6 publish/subscribe broker to run a migration command, **`strmqbrk`**. The second change is optional. You can modify existing or write new publish/subscribe programs to use the new programming interface. The changes are described in "Publish/Subscribe migration from Version 6.0" on page 367.

WebSphere Message Broker and WebSphere Event Broker Publish/Subscribe migration

The publish/subscribe broker in WebSphere Message Broker version 6.0 and 6.1, is replaced by using WebSphere MQ as the publish/subscribe broker. WebSphere Event Broker version 6.0 is replaced by WebSphere MQ. See Migrating publish/subscribe information to WebSphere MQ Version 7.0.1, "WebSphere Event Broker and WebSphere Message Broker migration tasks" on page 546, and "Differences between the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker version 6, and the publish/subscribe broker in WebSphere MQ version 7" on page 669

WebSphere Message Broker SCADA migration to WebSphere MQ Telemetry

The SCADA nodes in WebSphere Message Broker version 6.0 are no longer supported in version 7.0; see Changes to nodes in WebSphere Message Broker Version 7.0. You can migrate your SCADA applications to use a combination of WebSphere Message Broker version 7.0 and WebSphere MQ Telemetry; see "Telemetry migration from WebSphere Message Broker" on page 373.

WebSphere MQ migration concepts

Figure 63 on page 341 shows two runtime operating system environments. One environment is called `Server`, and contains a WebSphere MQ server and server application. The other is called `Client`, and contains a WebSphere MQ MQI client application. The server environment has one or more queue managers represented by **`QM`** using the installation of WebSphere MQ installed on the server.

The queue manager labeled QM-n? coexists on the same server as QM, but runs at a different release level. Multiple releases of WebSphere MQ installed in the same operating environment are called coexistent[2]. The WebSphere MQ installations for different release levels are not shown. Only z/OS supports multiple queue managers coexisting at different release levels in the same operating environment. The question-mark in the queue manager name indicates this capability might not be present in your environment.

Queue manager coexistence is important for migration in two respects:

1. It can be used to reduce the risk involved in migrating to a new command level, and reduce the downtime during the migration process.
2. You must consider any configuration implications of running some applications or clusters on the same server with queue managers at different command levels.

For details, see "Queue manager coexistence in Version 7.5" on page 345

The queue manager, QM*, represents queue managers of various levels installed on other servers.

[2] It is not necessary, but it is usual, for coexistent installations to be at different release levels.

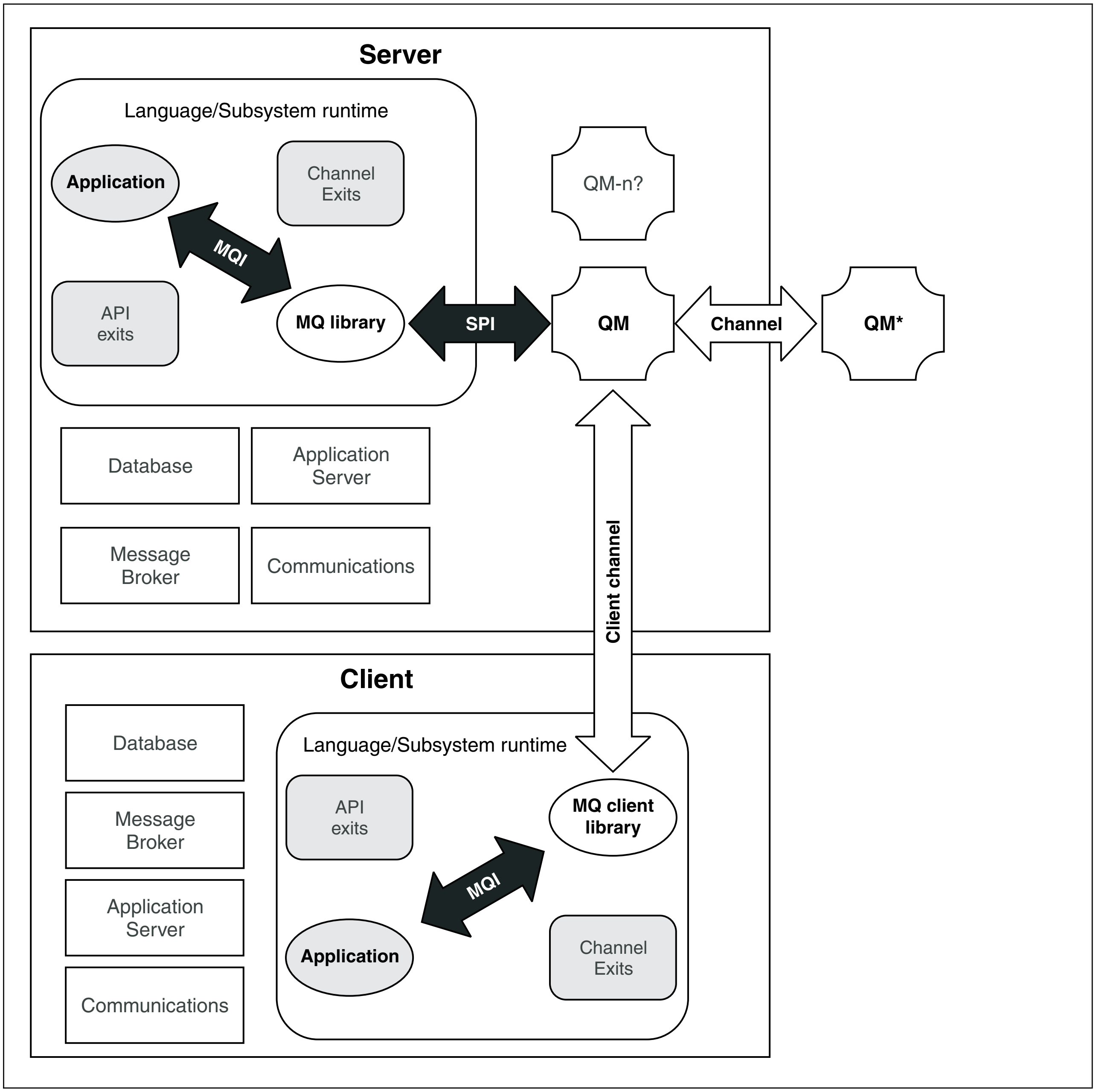


Figure 63. WebSphere MQ application migration model

New function in maintenance level upgrades

On platforms other than z/OS, IBM might introduce new functions between releases in maintenance level upgrades such as fix packs. A maintenance level upgrade including new function increases the maximum command level of an installation. When you apply the maintenance level upgrade, the installation supports the new command level. A queue manager associated with the installation is not able to use the new function until you have run **strmqm** to set the queue manager to the new command level.

A fix pack introducing a new function also includes regular fixes as a normal fix pack. If you do not want to use the function, but apply the fixes, do not enable the new function for any queue manager. The fix pack then is a fully reversible maintenance level upgrade. If you want to use a new function with a queue manager, you must change the command level of the queue manager. If one queue manager associated with an installation enables the new function, the fix pack introducing the new function and all previous fix packs cannot be removed anymore.

By applying a maintenance level upgrade that introduces a new function the IBM WebSphere MQ installation supports an additional command level. An installation can therefore support multiple command levels and multiple queue managers associated with it can run on different command levels.

To enable a new function for a queue manager, you must set the command level of queue manager to the command level supporting the new function. Because command levels are cumulative, the command level also supports all other functions introduced by previous maintenance level upgrades. By enabling a new function from a maintenance level upgrade you therefore enable all new functions from previous maintenance level upgrades, too.

Example

You have installed two fix packs including new functions:

1. Fix pack 7.1.0.2, introducing the command level 711.
2. Fix pack 7.1.0.15, introducing the command level 712.

You may have also installed other fix packs, but these did not introduce new function.

You want to use the new function from fix pack 7.1.0.15 on one of your queue managers. You enable command level 712 for that queue manager using the **strmqm** command. This enables all of the function introduced in both fix packs. It is not possible to use the function introduced by command level 712 without also enabling the function introduced by command level 711.

Once you have enabled command level 712 for a queue manager, the minimum version of product code to start the queue manager is 7.1.0.15. If you uninstall fix pack 7.1.0.15 from the installation with which the queue manager is associated, you will not be able to start the queue manager. In this situation, the **strmqm** command fails and issues error message AMQ7204 indicating that the queue manager has already been started by a newer release. If you have multiple installations of WebSphere MQ, you can start the queue manager using another installation which satisfies this minimum version requirement.

Configurable certificate validation policy on Windows and UNIX platforms

From fix pack 7.1.0.2 onwards, IBM WebSphere MQ can be configured to specify which SSL or TLS certificate validation policy is used to validate digital certificates received from remote partner systems.

If you need to ensure that your certificate validation is compliant with RFC 5280 for improved security in certificate validation, see Configuring certificate validation policies in WebSphere MQ for more formation.

See Certificate validation policies in WebSphere MQ for more information about IBM WebSphere MQ certificate validation policies.

Related tasks:

"Migrating queue managers to new-function fix packs" on page 703
This scenario illustrates running different levels of queue manager from a single installation using new-function fix packs. New function fix-packs are available on platforms other than z/OS. It contrasts migrating a queue manager to new command levels in new-function fix packs, to migrating a queue manager to a new command level in a new release. The scenario explains the relationship between new-function fix packs and maintenance fix packs.

Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS

You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

On different platforms, you employ different mechanisms to install and maintain software releases. Installing a release at a new maintenance level, and applying maintenance level upgrades to upgrade an earlier release to the same maintenance level, have different results.

If you start a queue manager running on a later release level, then migration of the queue manager to that release level is required. The migration tasks you must perform to migrate from one release to another are documented in “Migration tasks” on page 428. A list of all the changes, by migration path, are listed in “List of changes by version, release, and maintenance level” on page 554.

When you upgrade the maintenance or fix level of WebSphere MQ by applying a regular maintenance level upgrade, you can reverse the upgrade by removing the fix. When you upgrade the maintenance or fix level of IBM WebSphere MQ by applying a maintenance level upgrade containing a new function, you can reverse that upgrade and all previously reversible upgrades until a queue manager associated with the installation enables the new function.

On platforms other than z/OS, you cannot easily revert to a previous level of WebSphere MQ after installation. If you install a copy of WebSphere MQ obtained from Passport Advantage or from physical media, the installer uninstalls WebSphere MQ, if it is present. It then installs the new level of WebSphere MQ. To revert to the previous level of WebSphere MQ, you must keep the earlier installation image and any fixes you applied. Then you must uninstall the new level, reinstall the previous release level, and reapply the required fixes. If you have started any queue managers at the later level, they will not work with the restored level of WebSphere MQ[3]. To restore WebSphere MQ to its previous level, after starting any queue managers, you must first back up the queue managers. You can then restore the backup queue managers after restoring the previous level of WebSphere MQ.

Maintenance levels and fix levels are both supplied from the service site, Fix Central. Fix central has a function to tell you what upgrades you can apply to the current level of your system. WebSphere MQ Quick Beginnings tells you how to apply the maintenance level upgrade and back it out. If you back out a maintenance level upgrade, it returns WebSphere MQ code to the same level of code as before applying the maintenance level upgrade.

Characteristics of different types of upgrade

When you upgrade from one release to another, or apply maintenance refresh packs, fix packs, or interim fixes, the impact of the change depends on the extent of the change in VRMF level. The VRM codes are explained in “The version naming scheme for WebSphere MQ on platforms other than z/OS” on page 324.

Table 57 on page 344 describes characteristics of different upgrade paths for WebSphere MQ. In particular, notice that migration is required only if the version or release number changes. Other types of upgrade do not require migration, and are reversible if the upgrade is applied using a maintenance procedure.

[3] Unless you installed a later maintenance level upgrade, not a new release or version: then you could revert to an earlier maintenance level by reinstalling the earlier maintenance level upgrade. Queue manager data is compatible between maintenance levels.

Table 57. Types of upgrade (Platforms other that z/OS)

Upgrade type	Examples	Suggested testing	Type of installation	Manufacturing refresh	New features	WebSphere MQ Migration
Version	6.0, 7.0	Full testing of all applications	Full or upgrade installation	Yes	Yes	Yes
Release	5.3					
Modification	7.0.1	Regression test critical applications	Full, or upgrade installation, or maintenance package		Minor	No
Enabled new-function fix pack	No example yet	Regression test critical applications	Maintenance package & queue manager migration	No		
No-function or not enabled new-function fix pack	6.0.1.1, 7.0.1.3	Brief test of critical applications	Maintenance package		No	
Interim fix	7.0.0.1-WS-MQ-Windows-LAIZ50784	Test affected applications	Manual			

Related concepts:

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

WebSphere MQ operating environment migration

You might perform some migration tasks for WebSphere MQ as a result of upgrading the operating environment.

To find out what operating environment upgrades you must make before upgrading WebSphere MQ, compare the lists of supported software for different WebSphere MQ releases. For example, compare List of Supported Software for WebSphere MQ V6.0 with List of Supported Software for WebSphere MQ V7.0

If an operating environment change directly affects the migration to a new version of WebSphere MQ, it is listed in "List of changes by version, release, and maintenance level" on page 554. For example:

- "Windows: WebSphere MQ Installation affected by User Account Control on Windows systems" on page 668

- "Windows: SPX support on Windows Vista" on page 668

The change might affect WebSphere MQ migration indirectly. For example, the runtime linkage conventions for applications, or the way memory is allocated, might change.

Coexistence, compatibility, and interoperability

Coexistence is being able to install and run two or more versions of the same program on the same server. For WebSphere MQ, it normally means installing and running multiple versions of WebSphere MQ on a server. Compatibility is the ability to run applications from one level of queue manager with an earlier or previous level of the queue manager. It is also the ability to run client applications with different versions of the WebSphere MQ MQI client, and different levels of the queue manager. Interoperability is mainly the ability to exchange messages between different versions of WebSphere MQ. It can also mean the interoperability between others things, such as publish/subscribe brokers, or between components such as the WebSphere MQ classes for JMS and WebSphere Application Server.

Maintaining the compatibility, coexistence, and interoperability of WebSphere MQ is important in order to preserve the investment you make in applications and administrative procedures. Three areas to which this objective does not apply to as rigidly, are GUI interfaces, such as WebSphere MQ Explorer, information for service, such as FFST files and traces, and error messages. The text in an error message might change, to make the wording clearer or more accurate.

Queue manager coexistence in Version 7.5

Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

Single installation queue manager coexistence on all platforms

Single installation queue manager coexistence is useful in development and production environments. In development environments, you can set up different queue manager configurations to support different development activities. You can also work with multiple queue manager configurations on a single server, connected by channels, as if deployed on a network.

In production environments configuring multiple queue manager on a single server is less common. It has no performance or functional advantage over a single queue manager configuration. Sometimes, you must deploy multiple queue managers on server. It might be essential to meet the requirements of a particular software stack, governance, administration, or as a consequence of the consolidation of servers.

Multi-installation queue manager coexistence

Multi-installation[4] queue manager coexistence has always been supported on z/OS. It is new in Version 7.1 on UNIX, Linux, and Windows. With multi-installation queue manager coexistence on the same server you can run queue managers at different commands levels on the same server. You can also run multiple queue managers at the same command level, but associate them with different installations.

Multi-installation adds more flexibility to the coexistence of queue managers using a single installation. Any of the reasons behind running multiple queue managers, such as supporting different software stacks, might require different versions of IBM WebSphere MQ.

The biggest benefit of multi-installation identified by early users, is in upgrading from one version of IBM WebSphere MQ to another. Multi-installation makes upgrading less risky, less costly, and is more flexible in meeting the migration needs of applications running on a server.

[4] Do not confuse multi-installation queue manager coexistence with multi-instance queue managers. They are completely different, though they sound similar in English.

The key to migration flexibility is being able to install a new version alongside an existing installation; see Figure 64, which is extracted from "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401.

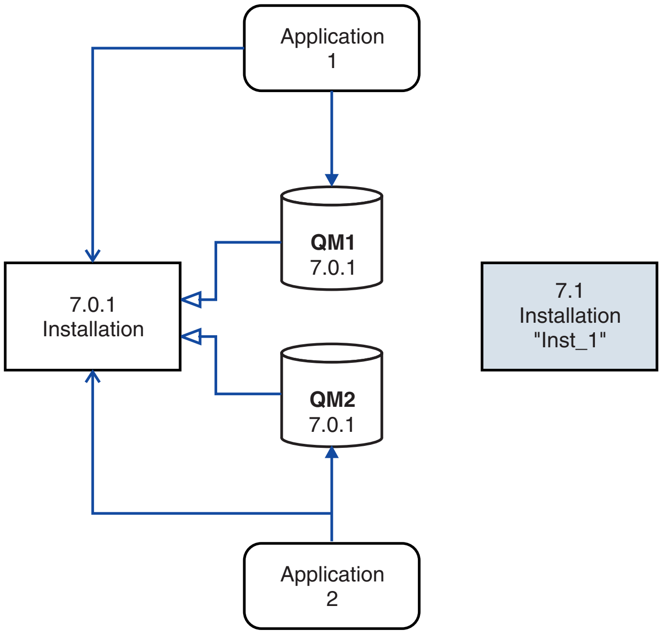


Figure 64. Side-by-side installation - step 2

When the installation is complete, and verified, migrate queue managers and applications to the new installation; see Figure 65. When migration is complete, uninstall the old installation.

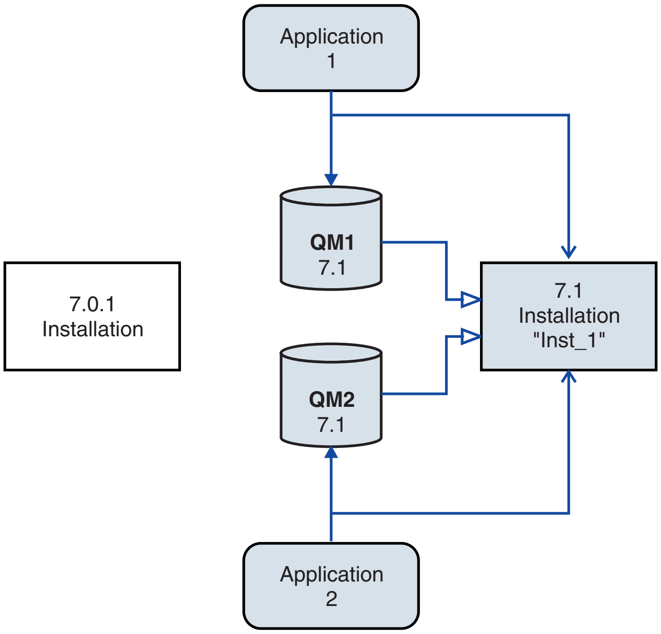


Figure 65. Side-by-side installation - step 4

Think of multi-installation as being the basis for a range of migration strategies. At one end is "Single-stage", in which you only have one installation on a server at a time. At the other end is multi-stage migration, in which you continue to run multiple installations at the same time. In the middle is side-by-side migration. Each of the three strategies is explained in these three tasks:

1. "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
2. "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
3. "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409

Another similar use of multi-installation is to support the migration of queue managers to a new fix level; see Figure 66. You maintain two installations, one of which has the latest fix pack applied, and the other has the previous maintenance levels. When you have moved all queue managers to the latest fix pack level, you can replace the previous fix pack with the next fix pack to be released. The configuration allows you to stage the migrating applications and queue managers to the latest fix pack level. You can switch the primary installation designation to the latest fix pack level.

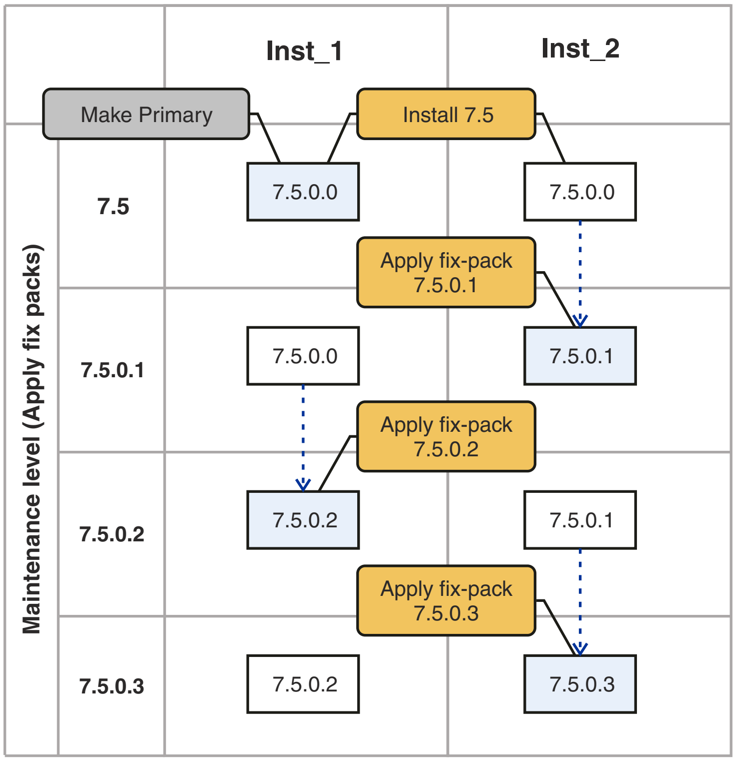


Figure 66. Rolling fix packs

Related concepts:

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows"
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks:

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

"UNIX, Linux, and Windows: Staging maintenance fixes" on page 718
Use multiple installations of IBM WebSphere MQ on the same server to control the release of maintenance fixes.

"Windows: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 485
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Multi-installation queue manager coexistence on UNIX, Linux, and Windows

You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Figure 67 shows two IBM WebSphere MQ installations, two queue managers, and three applications. Applications 2 and 3 are connected to `QM2`, and application 1 is connected to `QM1`. Applications 1 and 3 load IBM WebSphere MQ libraries from the `Inst_1` installation, and application 2 loads libraries from the Version 7.0.1 installation.

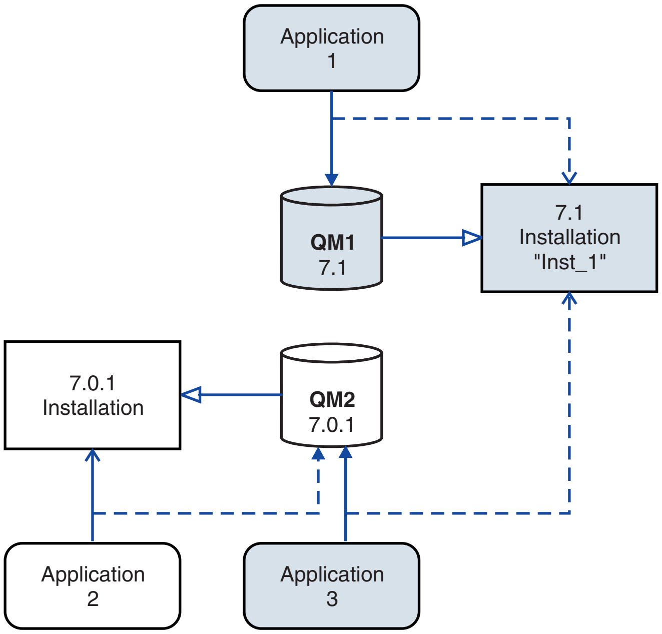


Figure 67. Coexistence of two queue managers using Version 7.0.1 and Version 7.1 installations

When you upgrade from Version 7.0.1 to Version 7.1, you can choose to run Version 7.0.1 alongside Version 7.1. The installation, illustrated in Figure 67, is called a multi-version installation. You can also install multiple copies of Version 7.1 alongside each other. That would be called multi-installation. Multi-installation is the more general term.

Apart from on z/OS, Version 7.0.1 did not support multi-installation. Before Version 7.1 becoming available, fix pack 7.0.1.6 was shipped with some fixes to make Version 7.0.1 compatible with Version 7.1 on the same server. With 7.0.1.6 installed, you can run one copy of Version 7.0.1 alongside multiple copies of Version 7.1. You do not have to apply the fix pack to upgrade Version 7.0.1 to Version 7.1 "in-place"; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394.

A multi-version installation that includes Version 7.0.1, does not behave the same way as a multi-installation that does not. The differences primarily affect how you might choose to configure how applications load IBM WebSphere MQ libraries, and run IBM WebSphere MQ commands. Because of these differences, think of the multi-version support provided in 7.0.1.6, as a migration aid to moving to a Version 7.1 multi-installation environment. The topics that explain the restrictions in Version 7.0.1 multi-version are listed in related links.

If you run multiple installations of IBM WebSphere MQ on a server you must consider three questions:

1. Which installation is a queue manager associated with; see "Queue manager association" on page 350?
2. Which installation does an application load; see "Loading IBM WebSphere MQ libraries" on page 350?
3. Which installation is a IBM WebSphere MQ command run from; see "Command association" on page 352?

Queue manager association

Before Version 7.1, queue managers on UNIX, Linux, or Windows were associated with the only installation on the server. With Version 7.1 installed on the same server as Version 7.0.1, you can change the association of a queue manager to Version 7.1 by running **setmqm**; see setmqm. You cannot change the association of a queue manager running a release of IBM WebSphere MQ earlier than Version 7.0.1 because you cannot install IBM WebSphere MQ Version 7.1 on a server with an installation of IBM WebSphere MQ earlier than Version 7.0.1.

A queue manager is permanently associated with an installation, until you choose to change the association with the **setmqm** command. You cannot associate a queue manager with an installation at a lower command level than the current command level of the queue manager.

In Figure 67 on page 349, `QM1` is associated with `Inst_1`. The association is made by running `setmqm -m QM1 -n Inst_1`. When `QM1` is first started, after running **setmqm**, if `QM1` was running Version 7.0.1, it is migrated to Version 7.1. `QM2` is associated with Version 7.0.1 because the association has not been changed.

Loading IBM WebSphere MQ libraries

The application connections to the queue managers are established by calling MQCONN or MQCONNX in the normal way.

Which IBM WebSphere MQ library an application loads depends on the configuration of the operating system loader and on the IBM WebSphere MQ installation the queue manager is associated with.

In Figure 67 on page 349, the operating system loads the IBM WebSphere MQ library from the `Inst_1` installation for applications 1 and 3. It loads the IBM WebSphere MQ Version 7.0.1 library for application 2. The operating system has loaded the wrong library for application 3. Application 3 requires the IBM WebSphere MQ Version 7.0.1 libraries.

Figure 68 on page 351 shows what happens to application 3. Application 3 is connecting to `QM2`, and `QM2` is associated with the IBM WebSphere MQ Version 7.0.1 installation. IBM WebSphere MQ detects that the operating system has loaded the wrong library to process calls from application 3 to `QM2`. IBM WebSphere MQ loads the correct library from the IBM WebSphere MQ Version 7.0.1 installation. It transfers the MQCONN or MQCONNX call to the IBM WebSphere MQ Version 7.0.1 library. Subsequent MQI calls that use the connection handle returned by MQCONN or MQCONNX, call entry points in the IBM WebSphere MQ Version 7.0.1 library.

Because IBM WebSphere MQ Version 7.0.1 libraries cannot load IBM WebSphere MQ libraries from other installations, there is no corresponding application in Figure 68 on page 351 that loads a IBM WebSphere MQ Version 7.0.1 library and connects to a queue manager running Version 7.1. If you attempt a connection to `QM1` with application 2, IBM WebSphere MQ returns an error; see 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE .

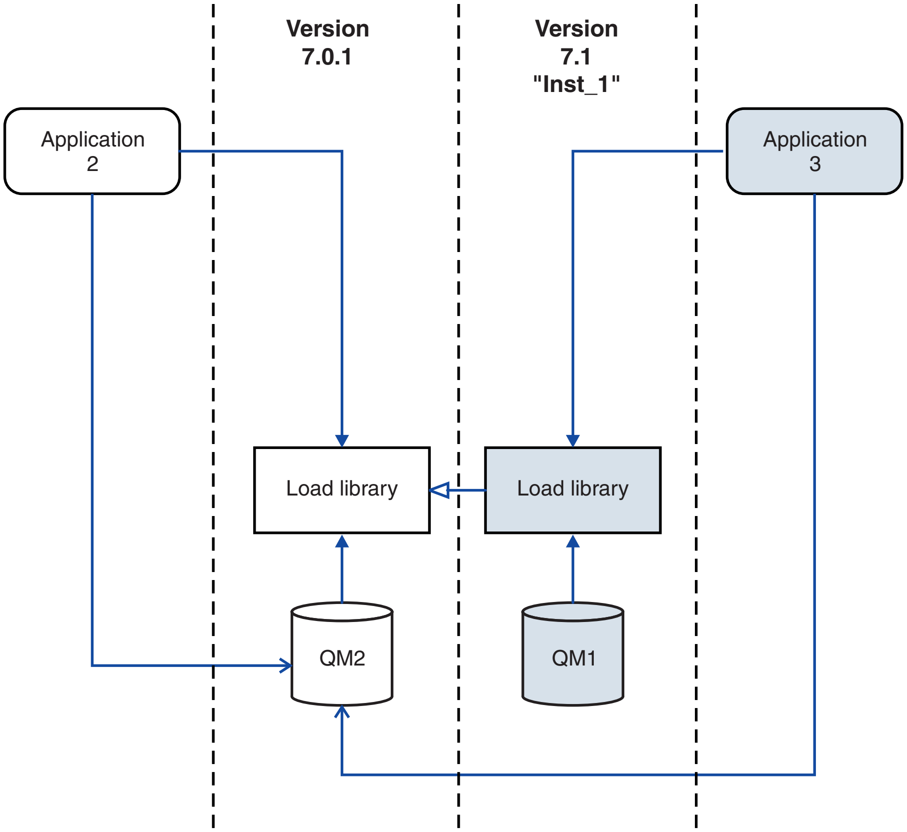


Figure 68. Loading calls in a different library

A Version 7.1 IBM WebSphere MQ library includes a routing capability that is based on the installation a queue manager is associated with. Earlier IBM WebSphere MQ libraries do not have a routing capability. The operating system can load a library from any Version 7.1 installation, or later, and IBM WebSphere MQ transfers MQI calls to the correct library.

The new loading capability of IBM WebSphere MQ libraries in Version 7.1 does not relax the restriction that an application compiled and linked at a later release level must not directly load a IBM WebSphere MQ library at an earlier release level. In practice the restriction is of less significance than in earlier releases, because as long as the operating system loads a library at the same or later level than the library the application was compiled and linked with, IBM WebSphere MQ can call any other level of IBM WebSphere MQ on the same server from Version 7.0.1 upwards.

For example, suppose you recompile and link an application that is to connect to a Version 7.0.1 queue manager using the libraries shipped with Version 7.1. At run time the operating system must load the Version 7.1 libraries for the application, even though the application connects to a Version 7.0.1 queue manager. IBM WebSphere MQ Version 7.1 detects the inconsistency and loads the Version 7.0.1 library for the application. The same applies to any future release. If the application is recompiled and linked against a later release, then the application must load a IBM WebSphere MQ library that matches the later release, even if it continues to connect to a Version 7.1 queue manager.

Your application might not be linked to a IBM WebSphere MQ library, but instead calls the operating system directly to load a IBM WebSphere MQ library. If the library that is loaded is from Version 7.1 or later, IBM WebSphere MQ checks the library is from the installation that is associated with the queue manager. If it is not, IBM WebSphere MQ loads the correct library.

Special migration considerations involving loading IBM WebSphere MQ libraries

You might have been asked to modify the installation of an earlier IBM WebSphere MQ release to satisfy the requirements of a build environment, or the IT standards in your organization. If you copied IBM WebSphere MQ libraries to other directories, or created symbolic links, you ended up with an unsupported configuration. The requirement to move IBM WebSphere MQ libraries to other directories was one of the reasons for changing the installation of IBM WebSphere MQ on UNIX and Linux. You can now install IBM WebSphere MQ into a directory of your choosing. You can also load IBM WebSphere MQ libraries from the `/usr/lib` directory, which is normally on the default load path on UNIX and Linux systems.

A common IT standard or build environment requirement is to include IBM WebSphere MQ libraries in the default load path on UNIX and Linux systems. IBM WebSphere MQ Version 7.1 has a solution. In Version 7.5 you can install IBM WebSphere MQ into a directory of your own choosing, and IBM WebSphere MQ can create symbolic links in `/usr` and its subdirectories. If you make a Version 7.1 installation primary by using the **setmqinst** command, IBM WebSphere MQ inserts symbolic links to the IBM WebSphere MQ libraries into `/usr/lib`. As a result, the operating system finds the IBM WebSphere MQ libraries in the default load path, if that includes `/usr/lib`.

Because IBM WebSphere MQ Version 7.1 libraries transfer calls to the correct installation, defining Version 7.1 installation as primary also results in the correct libraries being loaded for any application that is built with a link to `/usr/lib`, regardless of which queue manager it connects to. Unfortunately, this solution does not work if you have a Version 7.0.1 installation on the server, because then you cannot define a Version 7.1 installation as primary, and the Version 7.0.1 libraries do not load libraries from other installations. As an alternative to setting the Version 7.1 installation primary, use **setmqenv** with the `-k` or `-l` options to achieve a similar result.

You can find more information in Connecting applications in a multiple installation environment .

Command association

Examples of commands are **dspmqver**, **setmqinst**, **runmqsc**, and **strmqm**. The operating system must find a command in a IBM WebSphere MQ installation. Many commands also require a queue manager as an argument and assume the default queue manager, if a queue manager name is not provided as a parameter.

Unlike loading libraries, if a command includes a queue manager as a parameter, the command is not switched to the installation that is associated with the queue manager. You must use the **setmqenv** command to set up your environment correctly, so that any commands that you issue are run from the correct installation. You can provide a queue manager as a parameter to **setmqenv**, to set up the command environment for that queue manager; see Figure 69 on page 353.

On Windows, the **setmqinst** command sets global environment variables, and **setmqenv** local environment variables, including the `PATH` variable to find commands.

On UNIX and Linux, the **setmqinst** command copies symbolic links for a subset of the commands into `/usr/bin`; see External library and control command links to primary installation on UNIX and Linux. The **setmqenv** command sets up local environment variables, including the search path to the binary folder in the installation directory.

setmqenv must be on the search path in order to run it. One reason for having a Version 7.1 installation as primary is to be able to run setmqenv without having to configure the search path. If IBM WebSphere MQ Version 7.0.1 is installed on the server, no Version 7.1 installation can be primary and IBM WebSphere MQ Version 7.0.1 does not have a setmqenv command. The consequence is, you must provide a path to run the setmqenv command to set up the command environment for any of the Version 7.1 installations on the server.

Figure 69 shows two examples of running **setmqenv** to set up the command environment for the copy of IBM WebSphere MQ that is associated with the queue manager, QM1.

IBM WebSphere MQ for Windows Version 7.1

```
"MQ_INSTALLATION_PATH\bin\setmqenv" -m QM1
```

IBM WebSphere MQ Version 7.1 for UNIX and Linux

```
. MQ_INSTALLATION_PATH/bin/setmqenv -m QM1
```

Figure 69. Running **setmqenv**

Related tasks:

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

"UNIX, Linux, and Windows: Staging maintenance fixes" on page 718
Use multiple installations of IBM WebSphere MQ on the same server to control the release of maintenance fixes.

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

"Windows: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 485
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Related reference:

"Coexistence" on page 554
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information:

AIX linking 101

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Connecting applications in a multiple installation environment
On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

Dynamic-Link Library Search Order

External library and control command links to primary installation on UNIX and Linux
On UNIX and Linux platforms the primary installation is the one to which links from the /usr file system are made. However, only a subset of those links created with previous releases are now made.

Features that can be used only with the primary installation on Windows
Some Windows operating-system features can be used only with the primary installation. This restriction is due to the central registration of interface libraries, which might conflict as a result of multiple versions of IBM WebSphere MQ being installed.

Installation configuration file, mqinst.ini

Learn Linux, 101: Manage shared libraries

setmqenv

setmqinst

setmqm

strmqm -e CMDLEVEL

Application compatibility and interoperability with earlier versions of IBM WebSphere MQ

Connecting an application that is built against libraries shipped with a later version of IBM WebSphere MQ to an earlier version IBM WebSphere MQ is not supported. Avoid building applications against a later version, and redeploying them to a queue manager running at an earlier version, although some applications do work in practice.

IBM WebSphere MQ applications do interoperate with applications running on earlier versions of IBM WebSphere MQ, as long as they use no new function. IBM WebSphere MQ clients can connect to queue managers running at an earlier version than the client, as long as the client uses no new functions.

A IBM WebSphere MQ application that uses only functions provided by an earlier version of a queue manager can continue to send messages to the earlier version. It does not matter what version of IBM WebSphere MQ an application is built on and connected to. It can exchange messages with an application connected to an earlier version of IBM WebSphere MQ, as long as it does not use new function.

Consider these four cases; the first two cases are not supported though they might work in practice, the last two cases are supported. The first two cases require compatibility with an earlier version of IBM WebSphere MQ. The last two cases rely on the interoperability between all versions of IBM WebSphere MQ

1. Running a IBM WebSphere MQ server application, built with a later version of IBM WebSphere MQ, connecting to a queue manager running on a server with an earlier version of IBM WebSphere MQ installed.
2. Running a IBM WebSphere MQ client application, built with a later version of IBM WebSphere MQ, on a client platform with an earlier client installation, connecting to a queue manager running on a server with a later version of IBM WebSphere MQ installed.
3. Running a IBM WebSphere MQ client application, built with a later version of IBM WebSphere MQ, on a client platform with the later client installation, connecting to a queue manager running on a server with an earlier version of IBM WebSphere MQ installed.
4. Exchanging messages between a IBM WebSphere MQ client or server application, connected to a queue manager running on a server with a later version of IBM WebSphere MQ installed, with applications connected to a queue manager running on a server with an earlier version of IBM WebSphere MQ installed.

Plan to avoid the first two cases, as they are not guaranteed to work all the time. If you are running an incompatible configuration and you encounter a problem, you must rebuild your applications with the correct level of IBM WebSphere MQ. You can then continue with problem diagnosis.

Multi-installation and application loading

The new loading capability of IBM WebSphere MQ libraries in Version 7.1 does not relax the restriction that an application compiled and linked at a later release level must not directly load a IBM WebSphere MQ library at an earlier release level. In practice the restriction is of less significance than in earlier releases, because as long as the operating system loads a library at the same or later level than the library the application was compiled and linked with, IBM WebSphere MQ can call any other level of IBM WebSphere MQ on the same server from Version 7.0.1 upwards.

For example, suppose you recompile and link an application that is to connect to a Version 7.0.1 queue manager using the libraries shipped with Version 7.1. At run time the operating system must load the Version 7.1 libraries for the application, even though the application connects to a Version 7.0.1 queue manager. IBM WebSphere MQ Version 7.1 detects the inconsistency and loads the Version 7.0.1 library for the application. The same applies to any future release. If the application is recompiled and linked against a later release, then the application must load a IBM WebSphere MQ library that matches the later release, even if it continues to connect to a Version 7.1 queue manager.

Examples

1. You decide to rebuild a client application. Can you deploy it to your production environment that contains some earlier versions of client and server platforms?

   The answer is no, you must upgrade all the client workstations you deploy to, at least to the version of the client you have built. The queue managers running on earlier versions of IBM WebSphere MQ do not have to be upgraded. In practice all the clients are likely to work, but for maintainability you must avoid running incompatible levels of an application and the IBM WebSphere MQ client.
2. You deploy some IBM WebSphere MQ queue managers at a new version level. You have an existing IBM WebSphere MQ application that you use to send messages between the servers. Do you rebuild the application to deploy it onto the new servers? Can you deploy the old version onto the new servers?

   The answer is, either. You can continue to deploy the existing version of the application onto all your servers, or you can deploy the rebuilt application onto the new servers. Either configuration works. IBM WebSphere MQ supports running the existing application on later servers and sending messages from later application versions to earlier ones. What you must not do is to rebuild the application on the later version and redeploy it onto both the earlier and newer servers. IBM WebSphere MQ does not support compatibility with earlier versions.

Application compatibility and interoperability with later versions of WebSphere MQ

WebSphere MQ applications run against later versions of a queue manager without recoding, recompiling, or relinking. You can connect an application that is built against libraries shipped with an earlier version of WebSphere MQ to a queue manager running at a later version of WebSphere MQ.

If you upgrade a queue manager to a later version, existing applications built against its earlier version work without change. Exceptions are noted in “List of changes by version, release, and maintenance level” on page 554. Likewise applications connected to the WebSphere MQ Client, run against later versions of the client without recoding, recompiling, or relinking. You can deploy client applications built against earlier versions of the WebSphere MQ Client libraries to connect using later versions of the libraries.

All the following four cases are supported. The first two cases rely on the compatibility of later version of WebSphere MQ with applications built against earlier versions. The last two cases rely on the interoperability between all versions of WebSphere MQ.

You might change the operating environment as a prerequisite of migrating to a new level of queue manager. The operating environment changes, rather than changes in WebSphere MQ itself, might require application change, recompilation, or relinking. Sometime the operating environment change affects only the development environment, and the operating environment supports applications built at an earlier level. In which case, you might be able to run existing applications built at the older level of the operating environment. You might not be able to build any new applications until the operating environment is upgraded.

In the future, after you have migrated queue managers and clients to the latest release level, consider changing your applications to take advantage of new capabilities.

Compatibility between different versions of a WebSphere MQ MQI client and a queue manager

Any version and release of a WebSphere MQ MQI client can connect to any version and release of a WebSphere MQ queue manager. The MQI channel is automatically configured to the latest version that both the client and server support. If the client and server are different versions, the client application must use only the functions in the earlier version.

The compatibility between clients and queue managers applies only to the version and release (`V.R`) of the product. The statement of compatibility does not necessarily apply to the modification and fix pack level (`M.F`) of the product.

If there are known problems at a specific `V.R.M.F` of the product, an upgrade to a more recent fix pack for the same `Version.Release` is necessary.

When you upgrade a queue manager to a different version, you automatically upgrade WebSphere MQ libraries. The libraries are used by WebSphere MQ MQI client and server applications running on the same server as the queue manager. To access new functions from remote clients, you must also upgrade the WebSphere MQ MQI client installation on remote workstations. The WebSphere MQ MQI client includes the WebSphere MQ MQI client libraries.

Remote clients that have not been upgraded continue to work with an upgraded queue manager. The behavior of the client application might, in rare cases change. You must consult “List of changes by version, release, and maintenance level” on page 554, to find out whether the changes from one version to another affect your client applications.

Remote clients that are connected to upgraded queue managers can use the new functions in the release. If an upgraded remote client is connected to a queue manager that has not been upgraded, it must not use new functions. In rare cases, the behavior of the client might change; see “List of changes by version, release, and maintenance level” on page 554.

You can generally assume that upgrading the WebSphere MQ MQI client does not require you to recompile or relink the client application. You can also continue to use the same connection to the queue manager. If changes are required, they are identified in “Migration tasks” on page 428, for the particular migration path and platform you are concerned with.

The Client Channel Definition Table (CCDT) is an interface to customize the connection between a WebSphere MQ Client and a queue manager. Entries in the tables are client connections, which are defined using a queue manager. The version of a CCDT is the version of the queue manager used to define the client connections. If a WebSphere MQ MQI client uses CCDT to connect to a queue manager, the CCDT can be at a version greater than, less than, or equal to that of the client.

You can connect to a queue manager with an earlier WebSphere MQ Client or an earlier CCDT. If you are using a CCDT, and you plan to use new client channel configuration options, such as shared conversations, you must upgrade the CCDT, and therefore the WebSphere MQ MQI client installation to the new version.

Queue manager migration

After upgrading an installation, queue manager migration might be required. Migration takes place when you start a queue manager.

On other platforms, queue manager migration is always required for changes in the first two digits of the VRMF code. Changes in the maintenance and fix level, M and F in the VRMF code, never cause automatic queue manager migration. No migration was required for the upgrade from Version 7.0 to Version 7.0.1. The change from Version 7.0 to Version 7.0.1 did change the command level from 700 to 701. From Version 7.1 onwards, a change in the command level always requires queue manager migration, but if the change is shipped in a maintenance or fix pack, you have the choice of whether to increase the command level, and cause queue manager migration.

Command level always increases with a change in version or release. If you decide to use new function introduced in a maintenance level upgrade, you must change the command level. The converse is not the case. You do not have to change the command level when the fix level changes. You can decide to install the fix pack, but not use the new function. Whether or not you use the new function, the installation of the fix pack increases the maximum command level supported by the installation. Run the **dspmqver** command to display the current maximum supported command level.

Queue manager migration is the process of converting persistent queue manager data from one version to another. Persistent queue manager data includes log files and data in the queue manager directory. The data records changes to objects such as messages, subscriptions, publications, queue managers, channels, queues, and topics.

Queue manager migration is required and largely automatic.

After migrating to a new release the queue manager cannot be restored to an earlier release level.

See “Migration tasks” on page 428 for any queue manager changes you must perform manually.

You can reduce the downtime and risk caused by queue manager migration, by verifying the new version first, using a different queue manager. Unless the platform supports queue manager coexistence, you need to perform the verification on a different server, or in a virtualized environment on the same server.

If the platform you are upgrading supports queue manager coexistence, you can install the new version of WebSphere MQ on the same server, verify it, and minimize downtime to the time required to stop, backup, and restart the queue manager.

Note: If you are migrating a queue manager through multiple release levels, one level at a time, you must start the queue manager after each upgrade to migrate it. You must also start all the channels, to ensure they are migrated. For example, if you are upgrading from version 5.3 to version 7.0.1, you might choose to upgrade to version 6, and then to version 7.0.1; see "Migration paths" on page 320.

Related concepts:

"Migration paths" on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

Related tasks:

"Migrating a queue manager from Version 7.1 to Version 7.5" on page 429
The procedures for migrating a queue manager from Version 7.1 to Version 7.5 are listed by platform.

"Migrating a queue manager from Version 7.0.1 to Version 7.5" on page 458
The procedures for migrating a queue manager from Version 7.0.1 to Version 7.5 are listed by platform.

"Migrating a queue manager from Version 6.0 to Version 7.5" on page 492
The procedures for migrating a queue manager from Version 6.0 to Version 7.5 are listed by platform.

Related information:

dspmqver

Reverting a queue manager to a previous version

On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

On platforms other than z/OS, for which migration cannot be reversed, you must back up your system before starting migration. You can either back up queue manager data, or use a backup queue manager; see Backing up and restoring WebSphere MQ. To back up, you must stop the queue manager.

You can reduce the downtime and risk caused by queue manager migration, by verifying the new version first, using a different queue manager. Unless the platform supports queue manager coexistence, you need to perform the verification on a different server, or in a virtualized environment on the same server. If the platform you are upgrading supports queue manager coexistence, you can install the new version of WebSphere MQ on the same server, verify it, and minimize downtime to the time required to stop, backup, and restart the queue manager.

Related concepts:

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

Related information:

Backing up and restoring WebSphere MQ
Backing up queue managers and queue manager data.

WebSphere MQ MQI client migration

WebSphere MQ MQI client migration is the process of converting WebSphere MQ MQI client configurations, and client and server channels from one version to another. Client migration can take place after upgrading the WebSphere MQ MQI client. WebSphere MQ MQI client migrations are reversible.

Client migration on the client workstation is optional and manual. Client migration on the server is required and automatic. See "List of changes by version, release, and maintenance level" on page 554 for a list of any client changes. You must upgrade a WebSphere MQ MQI client before migrating a client workstation to make use of new configuration options. You can make configuration changes to client and server connection channels on the server, but they have no effect on a client workstation, until the client is upgraded.

An example of client migration performed at the client workstation is to manually migrate configuration settings to the `mqclient.ini` configuration file.

An example of client migration at the server is the automatic conversion of server-connection channels between Version 6.0 and Version 7.1 to use shared conversations. You must upgrade the client to Version 7.1 to use shared conversations.

An example of combined client and server migration is the deployment of a new client connection definition table (CCDT). To use a new version of the CCDT, generate the table on a queue manager that is at the new code level. Deploy the table to clients that are going to use it. To deploy the table to a client, you first must update the client to at least the same level as the queue manager that created the table.

WebSphere MQ MQI clients are interoperable with earlier and later versions of WebSphere MQ. Upgrading the WebSphere MQ MQI client makes new function available to client applications, and is important to maintain the service level. Migrating a WebSphere MQ MQI client gives it access to new configuration options, such as shared conversations.

The WebSphere MQ MQI client libraries, such as `mqic.dll`, are dynamic, and the application linkages to the libraries do not normally change. You do not relink a client application to pick up new WebSphere client libraries. The client picks up the new library next time the library is loaded by the client application. Do not move libraries from their installed directory. Linking to libraries in anything other than their installed directory is an unsupported configuration.

Related concepts:

"Application compatibility and interoperability with earlier versions of IBM WebSphere MQ" on page 354
Connecting an application that is built against libraries shipped with a later version of IBM WebSphere MQ to an earlier version IBM WebSphere MQ is not supported. Avoid building applications against a later version, and redeploying them to a queue manager running at an earlier version, although some applications do work in practice.

"Application compatibility and interoperability with later versions of WebSphere MQ" on page 356
WebSphere MQ applications run against later versions of a queue manager without recoding, recompiling, or relinking. You can connect an application that is built against libraries shipped with an earlier version of WebSphere MQ to a queue manager running at a later version of WebSphere MQ.

Related reference:

"MQI client: Client Channel Definition Table (CCDT)" on page 599
The client channel definition table has changed from Version 6.0 to Version 7.5. Existing clients must continue to use existing CCDTs. To use a Version 7.5 CCDT, you must update the client.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

Application migration and interoperation

WebSphere MQ supports running applications compiled and linked against previous versions of WebSphere MQ, with later levels of WebSphere MQ.

To migrate an application to run with a new level of WebSphere MQ, disconnect an application from the queue manager. Reconnect it when the queue manager is running again. However, it takes only one small difference in the interface between WebSphere MQ and the application to break an application, or make it behave wrongly. Sometimes a problem does not show up for a long time. For this reason, you must always test your applications against a new version of WebSphere MQ. The suggested extent of testing varies depending on the extent of the changes in WebSphere MQ; see "Characteristics of different types of upgrade" on page 343.

Application migration refers to four kinds of changes.

1. Application changes that are consequent to upgrading the operating environment along with the queue manager. Rarely, linkage conventions change. The most likely reason for a linkage change is switching from 32 bit to a 64 bit environment. If you are using SSL or TLS you might have to relink with a new secure library.
2. Changes that you must make to the application in order to run an application against a new level of queue manager. Changes of this sort are uncommon. However, you must check "List of changes by version, release, and maintenance level" on page 554 to see if any changes might affect your applications. Examples are the changes to JMS support between Version 6.0 and Version 7.5 listed in "JMS and Java changes" on page 613.
3. Changes that are not required, but you might want to make in the future, perhaps if there is some business reason to modify an application. An example is changing from using the command interface to publish/subscribe to using the call interface; see "Publish/Subscribe: Application migration" on page 634.
4. Changes to applications that are supplied by IBM, or other vendors, that require you to run migration utilities. The utilities convert the applications to running on the new version of WebSphere MQ. See "Publish/Subscribe migration from Version 6.0" on page 367 and "Migration of the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker" on page 369.

Do not load WebSphere MQ libraries from an earlier level. WebSphere MQ does not support connecting server applications loading libraries from the earlier level to connect to a later level of queue manager. On UNIX, Linux, and Windowsother platforms, the application load path must be set up to the location of the WebSphere MQ server libraries. You do not have to recompile and relink an application. Applications compiled and linked against an earlier version of WebSphere MQ can load libraries from a later version.

On UNIX, Linux, and Windows, from Version 7.1 onwards, WebSphere MQ loads the library from the installation the application is connecting to. An application must initially load a library of at least the same level as the application linked to. WebSphere MQ then loads the correct version of the library from the installation that the queue manager is associated with. If you have two installations of the same version, but at different fix levels, WebSphere MQ chooses which library to load. The choice is based on the queue manager the application is connected to. If an application is connected to multiple queue managers, it is possible that multiple libraries are loaded.

To help you write applications that can exchange messages with earlier versions of WebSphere MQ, WebSphere MQ provides data type versioning. Data type versioning assists you in exchanging messages that are compatible with target queue managers. A good programming practice is to set the version number of a data structure explicitly. Do not assume that the default version is the one you require. By setting the version explicitly, you are forced to look up what version to use. The description of the data type version tells you what level of queue manager supports that version.

It is poor practice to set the data type version to the current version. If you recompile your program against a new version of WebSphere MQ, the data type version might change with unexpected consequences.

Client applications are more likely to connect to different queue managers than applications written for a specific server. Plan carefully when writing an application that is to connect to different versions of a queue manager, and to queue managers on different platforms. The default values of some WebSphere constants, such as `MQPMO_SYNCPOINT`, `MQPMO_NO_SYNCPOINT` differ between platforms. Some functions are not available on all platforms.

You must be aware of, and code to, the capabilities of all the queue managers the application interacts with. It requires planning and design to write an application that works with different versions of a queue manager. There is no API provided with WebSphere MQ to restrict an application to a function subset common to the set of queue managers it interacts with. To improve interoperability, some developers choose to provide an MQI wrapper layer, or use MQI API exits, to control the functions programs use.

Related concepts:

"Application compatibility and interoperability with earlier versions of IBM WebSphere MQ" on page 354
Connecting an application that is built against libraries shipped with a later version of IBM WebSphere MQ to an earlier version IBM WebSphere MQ is not supported. Avoid building applications against a later version, and redeploying them to a queue manager running at an earlier version, although some applications do work in practice.

"Application compatibility and interoperability with later versions of WebSphere MQ" on page 356
WebSphere MQ applications run against later versions of a queue manager without recoding, recompiling, or relinking. You can connect an application that is built against libraries shipped with an earlier version of WebSphere MQ to a queue manager running at a later version of WebSphere MQ.

"Migration paths" on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

Queue manager cluster migration

You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

Cluster queue managers can participate in clusters with other queue managers running at different versions, which is why a staged migration is possible. Being able to stage a migration is important, as migrating each queue manager in a cluster takes time. By staging the migration, which leaves other queue managers that are in the cluster running, you reduce the effect of queue manager downtime on applications.

Migrate queue managers with full repositories first. Then migrate the other queue managers, which have partial repositories, one at a time. Complete migration of the entire cluster before starting to use new functions.

If you do have to start using new functions before completing migration of the entire cluster, you might need to refresh the partial repositories. After each migration of a queue manager with a partial repository, issue the **REFRESH CLUSTER** command on the newly migrated queue manager. The command updates the cluster records in the newly migrated queue manager, potentially receiving updates for any new attributes. Do not do this step if you migrated the entire cluster before using new function. The **REFRESH CLUSTER** command takes a long time for all the changes to work through the cluster.

Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

If full repositories are not migrated before partial repositories, the cluster continues to work, but without all the new features in a version working as expected. To work predictably, the full repository queue managers must be at the new command level to be able to store information from the rest of the cluster that arises from using new features.

For example, the information might be a new channel attribute, such as shared conversations, which were introduced in Version 7.0. Information about the shared conversation attribute of a channel between two other Version 7.0.1 queue managers can be stored in a version Version 7.0 full repository, but not in a Version 6.0 repository. If information about a channel with the shared conversation attribute is updated from the version 6.0 full repository, the definition loses its shared conversation attribute. “How mixed version cluster repositories are updated” explains how information is updated in a mixed-version cluster.

Note: If a queue manager is a member of a cluster, and is running at a release earlier than Version 6.0, you must migrate the queue manager to Version 6.0, Version 7.0, or Version 7.0.1, before migrating it to Version 7.5. You must start the queue manager after the first migration step, before proceeding to Version 7.5.

How mixed version cluster repositories are updated

Repositories store records for an object in a cluster in the version of the record format that matches the version of the queue manager hosting the repository. Repository queue managers forward object records, before they are stored, in the format that they are received in. The recipient ignores fields from a newer version, and uses default values for fields that are not present in the record.

Cluster repositories hold records that represent objects, for example, a queue record represents a cluster queue. A full repository holds records for all objects in the cluster. Partial repositories hold records for local objects and remote objects that are used locally. A repository record can hold information only about attributes at the same command level as the queue manager holding that repository. So for example, a Version 6.0 repository contains only Version 6.0 level attribute information. A Version 7.5 repository contains all Version 6.0 records, plus Version 7.5 records containing additional Version 7.5 attributes.

A repository stores a record it receives in its own version. If the record it receives is at a later version, the later version attributes are discarded when the record is stored. A Version 6.0 queue manager receiving information about a Version 7.5 queue manager stores only Version 6.0 information. A Version 7.5 repository receiving a version 6 record stores default values for attributes introduced in the version 7. The defaults define the values for the attributes that are not included in the record it receives.

A repository normally sends records in its own version format, which is the same as the format it has stored them in. There is one exception to this rule. When a full repository receives a record from a partial repository, it is immediately forwarded in the same format. So if a Version 6.0 full repository were to receive a record from a Version 7.5 partial repository, it would forward the Version 7.5 record. It sends the record to any other full repositories, and any other partial repositories that have subscriptions that match the record.

A partial repository reflects whichever full repository sent it the latest update to a record. As a consequence, you might see the information held by a Version 7.5 partial repository for new Version 7.5 attributes changing unexpectedly. The values might change from actual Version 7.5 information to default values. The changes occur if the full repositories in the cluster are at different levels. Migrate full repositories first to avoid instability.

A partial repository sends information about its objects to a full repository periodically at least once every 27 days. Information is sent about any object when it is altered or defined.

After migrating all full repositories to Version 7.5, some attributes might hold default values. The attributes might hold default values in place of actual values, if a repository has not received an update. You can refresh the repository in either of two ways:

- Alter the object which the record containing default values represents, for example, using `ALTER QL` for a local queue. The alteration forces the local repository to send the record again.
- Issue the **REFRESH CLUSTER** command on the partial repository which holds the record containing default values. **REFRESH CLUSTER** forces the partial repository to discard the record containing default values and get a new record as required.

  Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

In summary, for the most predictable, and fastest migration, when you stage cluster migration do these steps in the following order:

1. Migrate the queue managers with full repositories.
2. Migrate the queue managers with partial repositories.
3. Start using new function in the cluster.

Related information:

How long do the queue manager repositories retain information?
Queue manager repositories retain information for 30 days. An automatic process efficiently refreshes information that is being used.

Queue-sharing group migration

You can combine queue managers from different releases in a queue-sharing group. Limit the time you manage a mixed group to only as long as it takes to migrate all the queue managers to the same command level. You cannot combine a queue manager at Version 7.0 or later in the same queue-sharing group as queue managers earlier than Version 6.0. You must update all queue managers in a queue-sharing group with a coexistence PTF, before migrating any of them.

When you migrate queue managers in a queue-sharing group, aim to migrate all the queue managers to the new version as soon as you can. Queue-sharing groups can contain queue managers with a restricted mixture of versions. A mixture of queue managers in a queue-sharing group is supported so that you can migrate and test the upgrade of each queue manager. Migrate each queue manager, one at a time, leaving the queue-sharing group running. Mixed groups are harder to administer, than if all the queue managers are at the same version.

Example: Migrating version queue managers from Version 6.0 to Version 7.1

The following steps are a general guide how to migrate Version 6.0 queue managers that are part of a queue-sharing group to Version 7.1. See z/OS(r): Migrating queue sharing groups from version 6.0 to version 7.1 for detailed instructions.

1. Migrate any queue managers you are planning to be in the queue-sharing group, and not at Version 6.0, to at least Version 6.0.
2. Apply the Version 7.1 coexistence PTF to all the Version 6.0 queue managers in the group, or planned to be in the group.
3. Migrate Db2 tables. This step is required from Version 6.0 to Version 7.1. You can do this step one queue-sharing group at a time, or after you have migrated all queue-sharing groups.
4. Consider whether to turn on mixed-case security on Version 7.1 queue managers.
5. Migrate each Version 6.0 queue manager in the group to Version 7.1, setting **OPMODE** to `(COMPAT, 71)`.

   Tip: `DISPLAY SYSTEM` shows **OPMODE** as `(COMPAT, 6.`*`R.M`*`)` for the migrated queue managers, indicating you can revert them to version 6.
6. Verify the operation of each queue manager as you migrate it. If there is a problem, you can revert the queue manager to the version 6 command level.
7. Satisfy yourself that all the queue managers are working correctly in compatibility mode at the new command level.
8. Switch **OPMODE** to use the new command level; for example, set it to`(NEWFUNC, 71)`.

Related tasks:

"Migration tasks" on page 428
Follow these tasks to migrate queue managers from one version to another, and to migrate queue managers that are in clusters. You will also find assistance in migrating publish/subscribe or telemetry from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ.

Related reference:

"z/OS: Coupling Facility restrictions in a mixed queue-sharing group" on page 557
The migration of queue managers in a queue-sharing group, and coupling facilities, to a new level must be coordinated. Until all the queue managers in a group are at the new level, you are restricted in actions you can perform on the coupling facility.

"z/OS: MQSC commands in a mixed queue-sharing group" on page 557
Existing **MQSC** commands using new keywords and attribute values can be entered for routing to a migrated queue manager. You can enter the commands on any queue manager. Route the commands using **CMDSCOPE**. Commands with new keywords and attribute values, or new commands, routed to a previous version of queue manager, fail.

"z/OS: Mixed case RACF in a queue-sharing group" on page 557
How you turn on mixed-case RACF® support varies according to whether you have activated queue manager level checking or queue sharing group level checking, or both.

"z/OS: PCF commands in a mixed queue-sharing group" on page 557
On z/OS, create PCF commands only on a version 6.0 or higher queue manager. Also, if a command uses the parameter `MQCACF_COMMAND_SCOPE` to define a target queue manager, the target queue manager must be version 6 or later

"z/OS: Properties of objects in a mixed queue-sharing group" on page 557
Attributes that did not exist in earlier versions can be created and altered on queue managers of a later version in a mixed queue-sharing group. The attributes are not available to queue managers in the group that are at an earlier level.

"z/OS: Queue sharing group coexistence" on page 556
A queue-sharing group can contain queue managers running on IBM WebSphere MQ Version 6.0, and on later releases. The queue managers can access the same shared queues and other shared objects. A Version 6.0 queue-manager must have the coexistence PTF applied for the latest release. You cannot combine a queue manager in a queue-sharing group from a version earlier than Version 6.0 with a queue manager from a later release.

Migrate a queue manager in a high availability configuration

Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

High availability configurations of queue managers can increase the availability of IBM WebSphere MQ applications. If a queue manager, or server fails, it is restarted automatically on another server. You can arrange for IBM WebSphere MQ MQI client applications to automatically reconnect to the queue manager. Server applications can be configured to start when the queue manager starts.

High availability configurations on platforms other than z/OS are implemented either by using a high availability cluster solution or by using multi-instance queue managers. Red Hat Cluster Suite or Microsoft Cluster Service (MSCS) are examples of high availability cluster solutions.

The overall principles involved in queue manager migration in a high availability configuration are the same, whether you are migrating a multi-instance queue manager or a high availability cluster. In either case, the principles are as follows:

1. You must not restart a queue manager at a lower command level than the one it was previously running.
2. You cannot upgrade the code an active queue manager is running.
3. You cannot back up an active queue manager.

Overall steps to migrate a queue manager in a multi-instance queue manager configuration

The following terms are relevant:

active queue manager instance
A queue manager instance that has been started permitting standby instances, and is running.

standby queue manager instance
A queue manager instance that has been started permitting standby instances, and is in standby. It is ready to take over from the active instance automatically.

Base your migration procedure on the following steps.

1. If a multi-instance configuration is using a common location for WebSphere MQ code, you must create a second location for the upgraded code.
2. Before you start the migration process, create a different queue manager on a server on which you have installed the upgrade. Test the upgrade by performing whatever verification checks that your organization requires.
3. If you have a pool of servers that you pick from when starting a queue manager instance, upgrade WebSphere MQ on the servers that are in the pool and are neither active or acting as a standby.
4. Stop the standby queue manager instance. Make sure that you have no system management procedure running that restarts the instance automatically.

5. If you do not have a pool of servers, upgrade WebSphere MQ on the server that was running the standby instance.
6. Decide whether downtime or recoverability is more important in the migration:

   Follow these steps if recoverability is more important, and you must take a backup:

   a. Stop the active queue manager instance, without switching to any standby.
   b. Back up the queue manager.
   c. Start a queue manager instance, permitting standbys, on one of the upgraded servers.
   d. If you have a pool of upgraded servers, start another one, permitting standbys.

   If availability is more important, follow this procedure; you do not take a backup.

   a. Start a queue manager instance as a standby on one of the upgraded servers.
   b. Stop the active queue manager instance, switching to the standby.
   c. If you have a pool of upgraded servers, start another one, permitting standbys.
7. Upgrade the WebSphere MQ code on the server that was the active queue manager instance, and start it as the standby instance if you have not already started a standby.

Overall steps to migrate a queue manager in a high availability cluster

The following terms are relevant:

active server
The running server or active queue manager instance

passive server
A server that is ready to take over from the active server automatically.

inactive server
A server that is not prepared to take over automatically. The server might have been removed from the cluster, or taken offline in some way.

Base your migration procedure on the following steps. The details depend on the specific commands in the cluster concerned.

1. Before you start the migration process, create a different queue manager on a server on which you have installed the upgrade. Test the upgrade by performing whatever verification checks that your organization requires.
2. If you have four servers available, you can form two cluster pairs.

   With two pairs, the queue manager can continue to run in a cluster-pair at the old command level. When you are ready, you can transfer the queue manager to the pair of servers at the new command level.
3. Remove a passive server from the cluster. Make sure that the cluster cannot automatically restart the server. The server is made inactive.
4. If a high availability cluster is using a common location for WebSphere MQ code, you must create a second location for the upgraded code.
5. Install, or upgrade, WebSphere MQ code using the server that is not now running the queue manager.
6. Verify the upgrade by creating a different queue manager on the server, and performing whatever verification checks that your organization requires.
7. If more than half the servers remain in the cluster, remove a server, upgrade WebSphere MQ, and verify the upgrade. Each server is made inactive as part of the process. Continue until half the servers are upgraded.
8. If your active server is part of a remaining cluster, deactivate the passive servers so that the cluster cannot reactivate them automatically.
9. Decide whether downtime or recoverability is more important in the migration:

Follow these steps if recoverability is more important:

a. Stop the queue manager and remove the server from the cluster.
b. Back up the queue manager.

Or this step, if downtime is more important:

a. Add the migrated servers back into the cluster, as passive servers. Then switch the remaining server in the high availability server cluster over to one of the passive servers. The switch causes the running queue manager to stop, restarts it on one of the passive servers.

10. Upgrade any remaining high availability servers, and add them back into the cluster.

Related tasks:

"Windows: Migrating an MSCS configuration" on page 536
Migrate queue managers in MSCS configuration one node at a time, following these instructions.

Publish/Subscribe migration from Version 6.0

Connections between the MQTT client and the queue manager are always initiated by the MQTT client. The MQTT client is always the SSL client. Client authentication of the server and server authentication of the MQTT client are both optional.

In IBM WebSphere MQ Version 6.0, applications perform publish and subscribe operations by placing special request messages on certain queues. The IBM WebSphere MQ Version 6.0 publish/subscribe broker then reads and acts on these messages, for example by publishing messages to subscribing applications. State information, such as who is subscribing to which publications, is owned and maintained by the publish/subscribe broker. The broker is started and stopped independently from the queue manager.

In later versions of IBM WebSphere MQ, integrated publish/subscribe applications do not communicate with the broker in order to publish or subscribe. They use the new publish/subscribe API directly. The MQI call, MQPUT, is used to publish messages to a topic and MQSUB is used to subscribe. The queue manager itself performs the publish/subscribe function, so no separate publish/subscribe broker is required.

When you upgrade a queue manager from IBM WebSphere MQ Version 6.0 to a later version, the publish/subscribe broker is not upgraded. State information must be migrated from the WebSphere MQ publish/subscribe broker into the queue manager. Data that is migrated includes subscriptions, retained publications, hierarchy relations, and authorities. You migrate publish/subscribe resources from the broker to the queue manager by using the **strmqbrk** command, which previously started the publish/subscribe broker.

Publish/subscribe hierarchies and clusters

IBM WebSphere MQ Version 6.0 publish/subscribe brokers are connected into hierarchies so that publications and subscriptions can flow between them. After migrating, using **strmqbrk**, these hierarchies continue to function in later versions. IBM WebSphere MQ also contains a new method of allowing publications and subscriptions flow between queue managers; publish/subscribe clusters. To migrate to a publish/subscribe cluster, first migrate the queue managers in the hierarchy to a later version using **strmqbrk**. After checking publish/subscribe works, convert to using a cluster by creating cluster topics and altering the parent/child relations.

Publish/subscribe clusters are the same as queue manager clusters. You can use a queue manager cluster for publish/subscribe and to route point-to-point messages. You might decide to create separate publish/subscribe clusters, with different queue managers, because you want to manage the clusters separately.

New publish/subscribe applications are more likely to use clusters than hierarchies. What advantages do clusters have over hierarchies that might prompt you to convert from a hierarchy to a cluster?

Robustness
The failure of a queue manager can affect the transmission of publications and subscriptions between other queue managers in the hierarchy. In a cluster, every queue manager is connected to every other queue manager. A failure in one queue manager does not affect the connection between two other queue managers.

Simpler channel administration
A hierarchy uses manually configured channel connections between queue managers in the hierarchy. You must maintain these connections, adding, and removing channels as queue managers are added and removed from the hierarchy. In a publish/subscribe cluster, queue managers are connected by automatically maintained cluster connections.

Ease of use
Cluster publication topics, subscriptions, and their attributes are replicated to every member of a cluster. You can list and modify cluster topics and subscriptions attributes using the IBM WebSphere MQ Explorer. Your changes are replicated to other members of the cluster.

Consistency
In general, you must not connect the same queue managers together with hierarchies and clusters. If you have decided that new publish/subscribe applications are to use publish/subscribe clusters, avoid queue managers being part of both a hierarchy and a cluster. You must convert existing hierarchies to clusters.

The principal reasons for continuing to use a hierarchy, despite the advantages of publish/subscribe clusters, are threefold:

1. **strmqbrk** performs the migration to a hierarchy automatically. To convert the hierarchy to a cluster, you must do several manual tasks in addition to running **strmqbrk**.
2. You do not need to complete the migration of the whole hierarchy, before resuming your publish/subscribe applications using a hierarchy. In contrast, the whole hierarchy must be converted to a cluster before resuming your publish/subscribe applications using a cluster.
3. You might want to compare the performance of publish/subscribe in a cluster to a hierarchy, before switching your production systems to using publish/subscribe clusters.

Treat the conversion of a Version 6.0 publish/subscribe hierarchy to using clusters as a two-stage process. First, migrate all the queue managers in the hierarchy, and then convert the hierarchy to a cluster.

Publish/subscribe application migration

Publish/subscribe application migration from Version 6.0 to use integrated publish/subscribe in later versions is not necessary. Queued publish/subscribe continues to work alongside integrated publish subscribe. It can work in two ways. Set the **PSMODE** queue manager attribute to `ENABLED` for the queue manager to process queued publish subscribe command messages. If you set **PSMODE** to `COMPAT`, the queue manager runs integrated publish/subscribe, but not queued publish/subscribe. Another broker, such as WebSphere Message Broker, can process the queued publish/subscribe command messages while connected to the same queue manager.

Note: Publications and subscriptions cannot be exchanged between WebSphere Message Broker version 6.0 or 6.1, or WebSphere Event Broker version 6.0 and versions of IBM WebSphere MQ later than Version 6.0. Messages can be exchanged.

Queued publish/subscribe coexists on the same queue manager as integrated publish/subscribe, sharing publish/subscribe topics. Queued and integrated publish/subscribe also interoperate, exchanging subscriptions and publications between different queue managers.

When faced with a requirement to create another publish/subscribe application, or extend an existing application, which interface must you choose? You can choose either interface. The main reasons for continuing to develop a Version 6.0 application using the queued interface are as follows:

1. Continuing requirement to run the application on Version 6.0 queue managers.
2. Continuing requirement to run the application on version 6.0 or 6.1 WebSphere Message Broker or WebSphere Event Broker.
3. A small enhancement is required, and it is cheaper to extend the existing application than to migrate the existing application to use the integrated publish/subscribe interface, or to develop a new application to implement the enhancement.
4. The differences between the behavior of integrated and queued publish/subscribe mean that migration is a costly option.

The main reasons for developing new applications, and migrating existing applications to call the integrated publish/subscribe interface are as follows:

1. You find that the integrated publish/subscribe interface is easier to understand, and applications are quicker to develop.
2. You want to take advantage of clustering.

The topics in the section “Publish/Subscribe: Application migration” on page 634, explain how to convert queued publish/subscribe commands into integrated publish/subscribe calls. However, the publish/subscribe interfaces have different concepts of a publisher, subscriber, the identity of a subscription and publication, and the nature of streams, topics, and access control. Consequently, a straightforward mapping changing your application from creating a command message to calling a publish/subscribe function, works only for the simplest of applications.

You can combine queue publish/subscribe command messages and integrated publish/subscribe calls in the same application on the same queue manager connection. However, because the publication and subscription models are different, matching commands and calls do not necessarily produce the same results.

Related reference:

“Publish/Subscribe: Application migration” on page 634
The Version 6.0 IBM WebSphere MQ publish/subscribe command message interface has been superseded by the IBM WebSphere MQ publish/subscribe programming interface. It is your choice whether to migrate existing applications to the new interface. The following topics provide reference information to help you migrate applications to the publish/subscribe programming interface.

“Publish/subscribe changes” on page 633
Publish/subscribe function is now performed by the queue manager, rather than by a separate publish/subscribe broker. When you upgrade a IBM WebSphere MQ Version 6.0 queue manager, publish/subscribe function is not automatically migrated. You must migrate publish/subscribe using the **strmqbrk** command. After migration, with the differences noted in the following topics, existing Version 6.0 publish/subscribe programs work without change.

Migration of the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker

The publish/subscribe broker used in WebSphere Message Broker version 6.0 and 6.1 is replaced by the integrated publish/subscribe broker in IBM WebSphere MQ. IBM WebSphere MQ replaces WebSphere Event Broker version 6.0. The migration to IBM WebSphere MQ has two parts. The migration of publish/subscribe configuration data, and conversion of publish/subscribe collectives to publish/subscribe clusters.

Differences between the publish/subscribe brokers are described in “Publish/subscribe changes” on page 633.

Note: IBM WebSphere MQ does not support the content-based filtering provided in WebSphere Event Broker version 6.0. If you use this function and intend to continue to do so, you must migrate to WebSphere Message Broker version 7.0.

Chapter 9 of the IBM Redbook, contains a worked example of migrating the publish/subscribe broker to WebSphere MQ; see the related information.

The migration of the publish/subscribe configuration is carried out by **CSQUMGMB** on z/OS, and **migmqbrk** on other platforms.

The programs migrate the following publish/subscribe configuration data:
- Subscriptions
- Subscription points. Subscription points are supported only when MQRFH2 messages are used.
- Streams
- Retained publications

The task, "Migrating publish/subscribe configuration data from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1" on page 548, guides you through the migration of publish/subscribe configuration data. The migration is in three phases; see "Migration phases during the migration of WebSphere Message Broker or WebSphere Event Broker." It is partially automatic. You must set up access control on WebSphere MQ to reproduce the security environment on the broker. The migration procedure creates security commands for you to review and run to set up the access control on WebSphere MQ.

A WebSphere Message Broker and WebSphere Event Broker organize networks of publish/subscribe brokers that exchange publications and subscriptions. In WebSphere MQ, the matching topology is a publish/subscribe cluster. If you have defined publish/subscribe collectives, follow the task "Migrating a WebSphere Message or Event Broker publish/subscribe collective to a WebSphere MQ publish/subscribe cluster" on page 550.

Note:

You must migrate all the Message or Event Brokers in a collective to a IBM WebSphere MQ cluster at the same time. Any Message or Event Brokers that remain are not able to connect to the migrated brokers.

The WebSphere Event Broker or WebSphere Message Broker publish/subscribe information is not deleted by the migration process. This information is stored in the subscription database tables, and is available to use until you explicitly delete it.

Related information:

IBM Redbook: Connecting Your Business Using IBM WebSphere Message Broker V7 as an ESB

Migration phases during the migration of WebSphere Message Broker or WebSphere Event Broker

The migration of the publish/subscribe broker from WebSphere Message Broker or WebSphere Event Broker to WebSphere MQ has three phases. The rehearsal phase, the initial phase, and the completion phase.

The rehearsal phase

This phase creates a migration log, reporting any errors that might be found, but does not change the current configurations. You can use this phase to observe what the result of a real migration would be. Rehearsing the migration also produces a file containing security commands. The commands are suggestions to guide you setting up a security environment in the queue manager. The commands define a security environment for the queue manager that is equivalent to the one that existed in the broker. Review the security command file and determine the actions that you need.

The initial phase

This phase creates topic objects that might be needed in the queue manager. The topic objects are based on the Access Control List (ACL) entries that are defined in the broker. You must run this phase before you run the completion phase. The initial phase also produces a file containing security commands. The topic objects are created in anticipation of you running the security commands to create ACLs for the topic objects. Before you run the completion phase, you must review and modify the security command file as required and run the commands that you need. The initial phase also creates a migration log.

Note: If the migration process attempts to create a topic object, it first checks whether a suitable topic object exists in the queue manager. If it does, it uses that existing topic object. The check ensures that if the migration process is run multiple times, it does not attempt to create multiple topic objects for the same purpose.

If you modify the properties of one of the generated topic objects the migration process does not take account of the modification. For example, you might want to alter the wildcard property. The migration process uses the modified topic object, even though it no longer has the same properties that the migration process originally created for it. For this reason, with one exception, you must leave unchanged the topic objects that are created by the migration process until it has completed. The exception is you can change access permissions between the initial and completion phases,

The completion phase

The completion phase retrieves the current publish/subscribe definitions from the broker. It uses the definitions to create publish/subscribe definitions in the queue manager that is associated with the named broker. When the migration is complete, the queue manager publish/subscribe configuration is equivalent to the broker publish/subscribe configuration. The completion phase also creates a migration log containing the details of the migration.

The WebSphere Event Broker or WebSphere Message Broker publish/subscribe information is not deleted by the migration process. This information is stored in the subscription database tables, and is available to use until you explicitly delete it.

Publish/Subscribe Broker: Access Control List (ACL) migration

An automated process migrates the publish/subscribe broker from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ. The process does not migrate the ACLs automatically. The migration task provides two methods to migrate the ACL to WebSphere MQ.

The default ACL in WebSphere Event Broker and WebSphere Message Broker gives all user IDs access to any topic. In WebSphere MQ the default access control is for no user ID to have access to any topic. In the brokers, you can create ACL entries that deny access to topics. In WebSphere MQ, you can permit access only, you cannot deny access. Because of the differences in access control, the automated migration process cannot migrate ACLs from the brokers to WebSphere MQ.

The migration process solves this problem by generating a security command file. The file is produced by the rehearsal and initial phases of the migration process. It is a best attempt at creating security commands. The commands, when run, create similar access control for topics in the WebSphere MQ queue manager, as in the ACLs in the brokers.

If the migration process finds an ACL entry that denies access, it cannot produce a WebSphere MQ equivalent command. Instead, it reports it in the security command file and advises that the ACL migration must be performed manually. You must set up a security environment in the queue manager, equivalent to the one that existed in the broker. You must set up the environment in the queue manager before you run the completion phase of the migration.

To create an equivalent security environment in the queue manager, you can take one of two approaches. You can either modify the broker security settings to match the WebSphere MQ security approach, or modify the security command file as needed. If you decide to modify the broker security settings, run the rehearsal phase of the migration again to produce a new security command file. Continue to run the rehearsal phase until it stops reporting that the ACL migration must be performed manually.

Related reference:

"Publish/Subscribe Broker: ACL migration on Linux and UNIX platforms" on page 669
On Linux and UNIX platforms, it is possible for a username and groupname to have the same identifier. In which case the migration process is unable to determine whether an ACL applies to the username, the groupname, or both.

"Publish/Subscribe Broker: ACL wildcard subscriptions" on page 670
A subscription can use a wildcard scheme to subscribe to multiple topics. Access to one or more of the resulting matched topics might be denied by its access control list. The result in WebSphere MQ differs after migration from the result in WebSphere Event Broker and WebSphere Message Broker before migration.

"Publish/Subscribe Broker: Extra topic objects created by migration" on page 671
Administrative topic objects are created by the migration process. Some of the topic objects might be unnecessary. Although they do not cause any difference in behavior, deleting unwanted topic objects makes publish/subscribe easier to administer.

IBM WebSphere MQ Telemetry migration from Version 7.0.1

IBM WebSphere MQ Telemetry Version 7.0.1 is a separate feature that you install with IBM WebSphere MQ Version 7.0.1. Telemetry is integrated into IBM WebSphere MQ Version 7.5.

Existing IBM WebSphere MQ Telemetry applications are unaffected by the change. Telemetry channels are migrated to Version 7.5 when a queue manager is migrated to Version 7.5.

You can now administer IBM WebSphere MQ Telemetry from the command line and from applications, in addition to continuing to administer telemetry from IBM WebSphere MQ Explorer.

You can administer IBM WebSphere MQ Telemetry Version 7.0.1 only from the Version 7.0.1 IBM WebSphere MQ Explorer. If you connect the Version 7.5 explorer remotely to a Version 7.0.1 queue manager, no telemetry resources are displayed. You cannot connect a Version 7.5 IBM WebSphere MQ Explorer locally to a Version 7.0.1 queue manager on the same server.

With multiple installations, you can run IBM WebSphere MQ Telemetry Version 7.0.1 with IBM WebSphere MQ Version 7.0.1 on the same server as the integrated telemetry in Version 7.5.

To upgrade IBM WebSphere MQ Telemetry Version 7.0.1 from Version 7.5, you must uninstall IBM WebSphere MQ Telemetry from the Version 7.0.1 installation. You can uninstall IBM WebSphere MQ Telemetry Version 7.0.1 before or after you uninstall or upgrade IBM WebSphere MQ Version 7.0.1. Do not install the Version 7.5 telemetry component of the IBM WebSphere MQ Version 7.5 installation until you have uninstalled IBM WebSphere MQ Telemetry Version 7.0.1.

Related concepts:

"Migrating IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5" on page 455
Migrate IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5 by completing the tasks in this section. You must stop all IBM WebSphere MQ activity on the system before migrating.

Related reference:

"Telemetry: Installer integrated with IBM WebSphere MQ" on page 587
From Version 7.1, IBM WebSphere MQ Telemetry is no longer installed separately from IBM WebSphere MQ. It is installed as a component of IBM WebSphere MQ. If you installed IBM WebSphere MQ Telemetry with Version 7.0.1, you must uninstall it before installing the current version of the product.

Related information:

Administering WebSphere MQ Telemetry

WebSphere MQ Telemetry Version 7.0.1

Introduction to WebSphere MQ Telemetry

Telemetry migration from WebSphere Message Broker

The Supervisory Control and Data Acquisition (SCADA) input and output nodes on WebSphere Message Broker version 6.0, have been removed in version 7.0. The SCADA nodes are an interface to the MQ Telemetry Transport (MQTT) protocol, which is used to connect small devices and small message brokers over TCP/IP. To migrate applications that use the SCADA nodes, you must install WebSphere MQ Telemetry and change your message flows. You must also migrate any MQTT client applications to use version 3 of the MQTT protocol, if they are not already at version 3. You might be using a small message broker with your MQTT applications to connect to WebSphere Message Broker. In which case, you must migrate the small message broker to the WebSphere MQ Telemetry daemon for devices.

MQTT is a publish/subscribe protocol. WebSphere MQ now acts as a publish/subscribe broker for MQTT clients, including the WebSphere MQ Telemetry daemon for devices. The WebSphere MQ Telemetry daemon for devices is both a client and a server. To connect MQTT clients to WebSphere MQ, you must install and configure WebSphere MQ Telemetry. WebSphere MQ Telemetry is a WebSphere MQ service that provides a TCP/IP connection between an MQTT client and a queue manager. To configure WebSphere MQ Telemetry you use the WebSphere MQ Explorer.

To reconnect MQTT clients to the message flows that used SCADA nodes in WebSphere Message Broker version 6, you must replace the SCADA nodes. One approach is to replace the SCADA nodes with JMSInput and JMSOutput nodes.

On the JMSInput node, you can provide a subscription topic. Now that WebSphere Message Broker uses WebSphere MQ as its publish/subscribe broker, the subscription is managed by a WebSphere MQ queue manager. When an MQTT client publishes to a topic, WebSphere MQ Telemetry passes the publication to the queue manager the MQTT client is connected to. You might connect the MQTT client to the same queue manager as the broker. Alternatively, you can connect the MQTT client to a different queue manager, and connect both queue managers together in a publish/subscribe cluster. Whichever topology you choose, if the MQTT publication topic matches the JMSInput node subscription topic, the publication is delivered to the WebSphere Message Broker application.

To reverse the flow of a publication, from the WebSphere Message Broker to the MQTT client, use a JMSOutput node, or a Publication node. On the JMSOutput node, define the publication topic. A message output by the JMSOutput node is published on that topic to the WebSphere MQ queue manager the broker is connected to. If the queue manager has a matching subscription created by an MQTT client, the publication is forwarded to the MQTT client. If the MQTT client is connected to a different queue manager, the publication is forwarded using a publish/subscribe cluster.

Related tasks:

Migration of telemetry applications from using WebSphere Message Broker v6 to use WebSphere MQ Telemetry and WebSphere Message Broker v7

Migrating telemetry applications from using WebSphere Message Broker v6 to use IBM WebSphere MQ Telemetry and WebSphere Message Broker v7 requires a number of steps. Only the last step involves modifying the telemetry application. The first steps are migrating IBM WebSphere MQ and WebSphere Message Broker to v7.

Related information:

MQTT.org

Telemetry

WebSphere MQ Telemetry Transport

Planning IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5 migration by platform

Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

If there are concepts about migration you do not understand, read “Introduction to WebSphere MQ migration” on page 320 first.

AIX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.1, see Table 52 on page 322 in “Where to find a topic about a specific migration path” on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.1 and Version 7.5 on the same server.
3. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See “AIX: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 430.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.

8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.

   Consider whether the application must be able to run on both Version 7.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

➡ Fix Central

➡ IBM Passport Advantage

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

➡ WebSphere MQ - SupportPacs by Product

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

HP-UX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.1 and Version 7.5 on the same server.
3. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrate a queue manager in a high availability configuration" on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See "HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 434

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.

9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.

   Consider whether the application must be able to run on both Version 7.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.

10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.

11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM WebSphere MQ migration” on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

“IBM WebSphere MQ upgrade” on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details

vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Linux: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.1, see Table 52 on page 322 in “Where to find a topic about a specific migration path” on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.1 and Version 7.5 on the same server.
3. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See “Linux: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 437

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.

Consider whether the application must be able to run on both Version 7.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.

10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM WebSphere MQ migration” on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

“IBM WebSphere MQ upgrade” on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it

depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Solaris: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.1 and Version 7.5 on the same server.
3. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrate a queue manager in a high availability configuration" on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See "Solaris: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 442

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.

   Consider whether the application must be able to run on both Version 7.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.

10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.

11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM WebSphere MQ migration” on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

“IBM WebSphere MQ upgrade” on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Windows: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.1, see Table 52 on page 322 in “Where to find a topic about a specific migration path” on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.1 and Version 7.5 on the same server.
3. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See “Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 445

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.

   Consider whether the application must be able to run on both Version 7.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

Include the procedures and applications you identified in steps 6 and 7 in your regression tests.

11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Directory structure on Windows systems
How to find queue manager configuration information and directories on Windows.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM WebSphere MQ migration” on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

“IBM WebSphere MQ upgrade” on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Publish/Subscribe migration from version 6.0

“Migrating publish/subscribe from Version 6.0 to Version 7.5” on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

Planning IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 migration by platform

Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

If there are concepts about migration you do not understand, read "Introduction to WebSphere MQ migration" on page 320 first.

AIX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.0.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.

2. Decide whether to run Version 7.0.1 and Version 7.5 on the same server.

   "Multi-installation" is new for Version 7.1. You can run Version 7.0.1 in parallel with Version 7.5. You can model the migration of WebSphere MQ on one of these three migration scenarios. You will find advantages and disadvantages with each scenario.

   - "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
   - "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
   - "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409

3. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.

4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.

5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.

6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrate a queue manager in a high availability configuration" on page 365.

7. Plan to migrate your queue manager to Version 7.5.

   See “AIX: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 459.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 561.

   Consider whether the application must be able to run on both Version 7.0.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

HP-UX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.0.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.0.1 and Version 7.5 on the same server.

   "Multi-installation" is new for Version 7.1. You can run Version 7.0.1 in parallel with Version 7.5. You can model the migration of WebSphere MQ on one of these three migration scenarios. You will find advantages and disadvantages with each scenario.
   - "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
   - "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
   - "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
3. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.

6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See “HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 463

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 561.

   Consider whether the application must be able to run on both Version 7.0.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Linux: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.0.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.

2. Decide whether to run Version 7.0.1 and Version 7.5 on the same server.

   "Multi-installation" is new for Version 7.1. You can run Version 7.0.1 in parallel with Version 7.5. You can model the migration of WebSphere MQ on one of these three migration scenarios. You will find advantages and disadvantages with each scenario.

   - "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
   - "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
   - "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409

3. Review all the changes in WebSphere MQ that affect you.

See changes from “Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 561.

4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.

5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.

6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.

7. Plan to migrate your queue manager to Version 7.5.

   See “Linux: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 466

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.

8. Plan to update any manual or automated procedures you have written with changes to messages and codes.

9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 561.

   Consider whether the application must be able to run on both Version 7.0.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.

10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.

11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Solaris: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.0.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Decide whether to run Version 7.0.1 and Version 7.5 on the same server.

   "Multi-installation" is new for Version 7.1. You can run Version 7.0.1 in parallel with Version 7.5. You can model the migration of WebSphere MQ on one of these three migration scenarios. You will find advantages and disadvantages with each scenario.
   - "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394

- “UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version” on page 401
- “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409

3. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 561.
4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
7. Plan to migrate your queue manager to Version 7.5.

   See “Solaris: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 470

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 561.

   Consider whether the application must be able to run on both Version 7.0.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 7.0.1, see Table 52 on page 322 in "Where to find a topic about a specific migration path" on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.

2. Decide whether to run Version 7.0.1 and Version 7.5 on the same server.

   "Multi-installation" is new for Version 7.1. You can run Version 7.0.1 in parallel with Version 7.5. You can model the migration of WebSphere MQ on one of these three migration scenarios. You will find advantages and disadvantages with each scenario.

   - "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
   - "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
   - "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409

3. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.

4. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.

5. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.

6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrate a queue manager in a high availability configuration" on page 365.

7. Plan to migrate your queue manager to Version 7.5.

   See "Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 474

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.

8. Plan to update any manual or automated procedures you have written with changes to messages and codes.

9. Plan to update applications that might be affected by changes you identified in "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.

   Consider whether the application must be able to run on both Version 7.0.1 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.

10. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

    Include the procedures and applications you identified in steps 6 and 7 in your regression tests.

11. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

12. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Directory structure on Windows systems
How to find queue manager configuration information and directories on Windows.

Fix Central

IBM Passport Advantage

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Publish/Subscribe migration from version 6.0

"Migrating publish/subscribe from Version 6.0 to Version 7.5" on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version

Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

Before you begin

This scenario is one of three, which describe alternative ways to upgrade queue managers from an earlier version of the product. The other scenarios are as follows:

1. Install the latest version of the product alongside an earlier version ; see "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401.
2. Run the latest version of the product alongside an earlier version ; see "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409.

Read these three tasks to plan how you are going to migrate to the multi-installation environment of the latest version. Even if you do not plan to have more than one version of the installation on a server, read this topic. The steps to upgrade IBM WebSphere MQ from Version 7.0.1 to Version 7.5 have changed.

These topics are for planning multi-installation migration. The planning topics guide you in deciding what other tasks you must perform to migrate queue managers and applications to the latest version. For the precise sequence of commands to upgrade a queue manager to the latest version, do the migration task for the platform you are interested in. All the tasks are listed by platform in the links at the end of this topic. As part of the queue manager migration task, back up your existing queue manager data. Even on a multi-installation server, queue managers cannot be restored to a previous command level after migration.

About this task

In the single-stage migration scenario, the installation of the latest version of the product replaces an earlier version in the same installation location. It is the same migration process that you would have used to upgrade the product prior to IBM WebSphere MQ Version 7.0.1.6 . It is now termed "single-stage" migration, in contrast to "side-by-side" and "multi-stage" migration.

The advantage of single-stage migration is that it changes the configuration of a queue manager on the earlier version as little as possible. Existing applications switch from loading the libraries from the earlier version, to loading the libraries of the latest version, automatically.

Queue managers are automatically associated with the installation on the latest version. Administrative scripts and procedures are affected as little as possible by setting the installation to be the primary installation. If you set the installation of the latest version to be the primary installation, commands such as **strmqm** work without providing an explicit path to the command.

The description of the migration scenario starts with the example in Figure 70 on page 396.

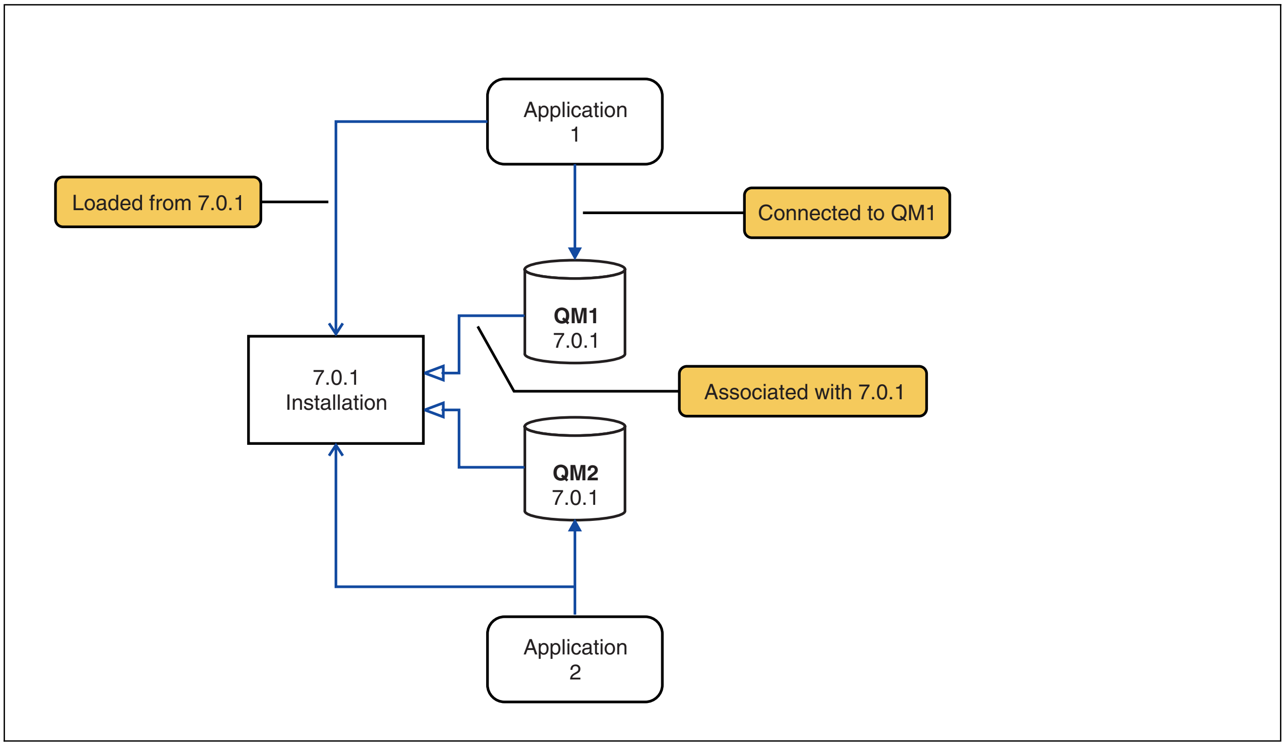


Figure 70. Existing server on an earlier version

Four types of object are important to consider during migration: installations, queue managers, administrative procedures, and applications. The diagram shows the installation an application loads IBM WebSphere MQ libraries from, connections between applications and queue managers, and associations between queue managers and installations. Administrative procedures are omitted from the diagram. Administrative procedures contain IBM WebSphere MQ commands, and scripts that use commands.

"Loading from 7.0.1" in Figure 70, refers to the IBM WebSphere MQ installation from which the application loads the IBM WebSphere MQ library it requires; see "Loading IBM WebSphere MQ libraries" on page 350. The connection is a call to MQCONN or MQCONNX, and has not changed from the earlier version of the product to the latest version. The association is the installation that the queue manager is associated with. The association is created either by running the **setmqm** command, or by starting a queue manager on the earlier version; see Associating a queue manager with an installation.

To run a command, the operating system must find the command in a IBM WebSphere MQ installation. For some commands, you must run the command from the installation that is associated with the correct queue manager. IBM WebSphere MQ does not switch commands to the correct installation. For other commands, such as **setmqinst**, you can run the command from any installation that has the latest version of the product installed.

If an earlier version of the product is installed, the command that is run is the command for that version, unless the search path is overridden by a local setting. You can override the search path by running **setmqenv**. If Version 7.0.1 is not installed, you must set the correct path to run a command. If you have set a primary installation, the command that is run is the copy in the primary installation, unless you override the selection with a local search path.

Procedure

1. Upgrade the earlier version of the product to the latest version in the same installation directory; see Figure 71 on page 397.

a. Decide on an installation naming convention. Give the installation a name of your choosing, or accept the default installation name. For the first installation, the default name is *Installation1*. For the second installation, the name is *Installation2*, and so on.

b. Upgrade the earlier version of the product to the latest version in place, or uninstall the earlier version, without deleting any queue managers, and install the latest version in the same default location. Whether you have to uninstall your previous version of the product depends upon your operating system.

   On:

   - AIX
   - Windows
   - IBM i, where the process is known as a *slip* install

   you do not have to uninstall a previous version of the product.

   On:

   - HP-UX
   - Linux
   - Solaris

   you have to uninstall the previous version of the product.

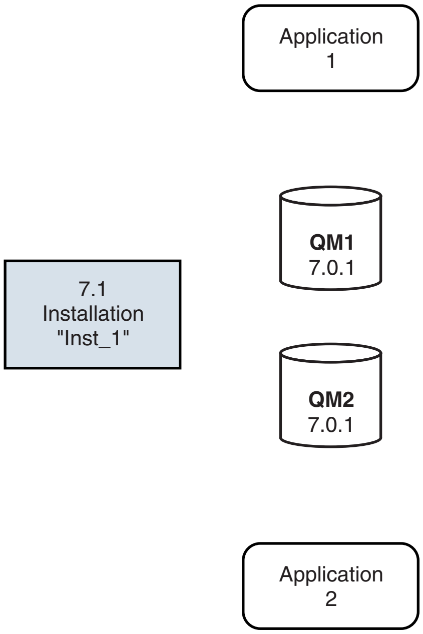


Figure 71. Upgrade of earlier version of product to latest version in the default directory

- A reason for installing into the same location is to simplify application migration. If you change the installation location, you might remove IBM WebSphere MQ libraries from an application search path. To migrate an application search path you must modify the application environment, or more rarely, the application itself.
- The default installation path is specified as a load path in the IBM WebSphere MQ build scripts for UNIX and Linux. After installation of the latest version, the load libraries of the latest version of IBM WebSphere MQ are in the same location as were the libraries of the earlier version. If you built applications by following the examples in the product documentation for the earlier version, the applications load the correct libraries in the latest version.

2. Make the latest version of the installation the primary installation; see Figure 72.
   a. Run the **setmqinst** command

On Windows

```
"Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
```

On UNIX

```
Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
```

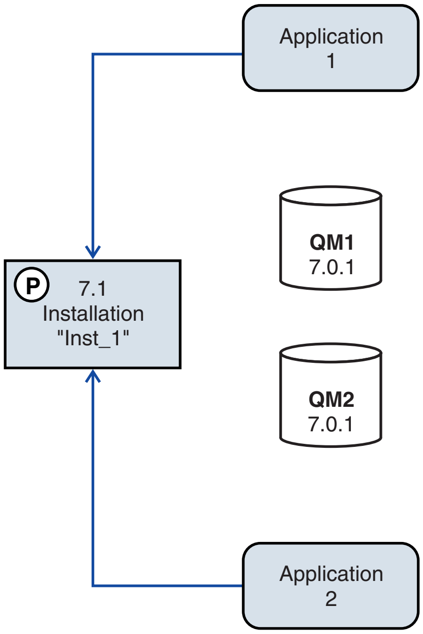


Figure 72. Prepare the installation and applications

- Make the installation primary to avoid specifying a search path to run IBM WebSphere MQ commands.
- If there is a primary installation, UNIX and Linux applications that expect to find the IBM WebSphere MQ library in /usr/lib, find a symbolic link to the library in /usr/lib/32[5]. /usr/lib/32 is normally in the default search path. It is also specified as a load path in the IBM WebSphere MQ build scripts for UNIX and Linux.
- It is sufficient to link applications only to /usr/lib. With a primary installation of the latest version of the product defined on the server, an application can connect to any queue manager associated with any installation on the server. IBM WebSphere MQ loads the correct library for the application.

3. Start the queue managers and applications; see Figure 73 on page 399.
   a. Optional: Run the **setmqm** command to associate the queue managers with Inst_1.

[5] /usr/lib for 64 bit applications.

```
setmqm -m QM1 -n Inst_1
setmqm -m QM2 -n Inst_1
```

Running **setmqm** to associate the queue managers with `Inst_1` is optional, because these queue managers are being started on the latest version of the product for the first time.

Windows If you have multiple installations, note that queue managers that were configured to start automatically, and remain after uninstalling IBM WebSphere MQ Version 7.0.1, automatically start under any other existing Version 7.1 (or laer) installation when either the machine reboots, or the Service for that installation is restarted. In order to avoid this, ensure that all queue managers have been moved to the desired installation before uninstalling IBM WebSphere MQ Version 7.0.1.

b. Run the **strmqm** command to start the queue managers and migrate them to the latest version of the product.

```
strmqm QM1
strmqm QM2
```

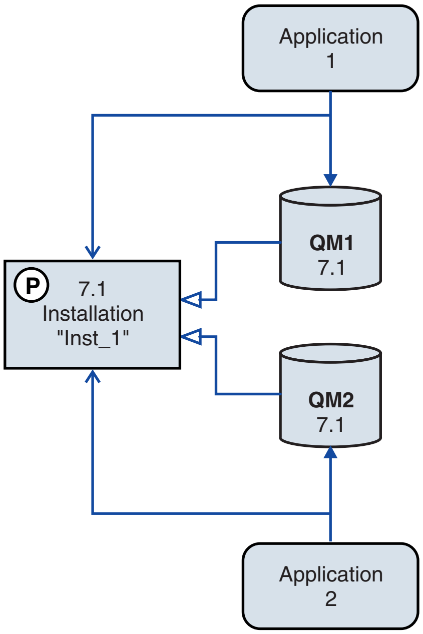


Figure 73. Restart the queue managers and applications

- When an application connects to a queue manager, the operating system searches its load path to load the IBM WebSphere MQ library[6]. A Version 7.5, or later, library contains code that checks that the queue manager is associated with an installation. If a queue manager is associated with a different installation, IBM WebSphere MQ loads the correct IBM WebSphere MQ library for the installation the queue manager is associated with.

[6] On Windows, the IBM WebSphere MQ library is a DLL. A DLL is sometimes called a load library or a shared library. The entry points to a DLL are defined in a link library, with the file extension `.lib32` or `.lib`. The `.lib` library is linked at build-time and the DLL loaded at runtime.

What to do next

You cannot reinstall an earlier version of the product on a system that has the latest, or any other, version of IBM WebSphere MQ installed.

Related concepts:

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks:

"Planning IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5 migration by platform" on page 374
Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

"AIX: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 430
Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

"Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 445
Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Related information:

Installing WebSphere MQ server on AIX
You can install IBM WebSphere MQ server on an AIX system.

Installing WebSphere MQ server on HP-UX

Installing WebSphere MQ server on Linux
You can install a IBM WebSphere MQ server on either a 32-bit or 64-bit Linux system.

Installing WebSphere MQ server on Solaris
You can install an IBM WebSphere MQ server on Solaris either interactively or silently.

Installing WebSphere MQ server on Windows
This topic describes how to install IBM WebSphere MQ server on Windows systems by using the Launchpad. This procedure can be used for installing a first or a subsequent installation.

Associating a queue manager with an installation
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

setmqenv

setmqinst

setmqm

UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version

Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

Before you begin

If you are using IBM WebSphere MQ Version 7.0.1, you must ensure that you are running IBM WebSphere MQ Version 7.0.1.6 before installing the latest version of the product on the same server. Go to Fix Central to obtain the fix pack.

This scenario is one of three, which describe alternative ways to upgrade queue managers from an earlier version of the product. The other scenarios are as follows:

1. Replace the earlier version with the latest version; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394.
2. Run the latest version of the product alongside an earlier version ; see “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409.

Read these three tasks to plan how you are going to migrate to the multi-installation environment of the latest version. The step-by-step migration scenario sits half-way between the single-stage and multi-stage migration scenarios.

These topics are for planning multi-installation migration. The planning topics guide you in deciding what other tasks you must perform to migrate queue managers and applications to the latest version. For the precise sequence of commands to upgrade a queue manager to the latest version, do the migration task for the platform you are interested in. All the tasks are listed by platform in the links at the end of this topic. As part of the queue manager migration task, back up your existing queue manager data. Even on a multi-installation server, queue managers cannot be restored to a previous command level after migration.

About this task

In the “side-by-side” migration scenario, you install the latest version of IBM WebSphere MQ alongside queue managers that continue to be associated with Version 7.0.1, or later.

When you are ready to migrate the queue managers, and applications, to the latest version:

1. Stop all the queue managers.
2. Uninstall the earlier version of the product.
3. Migrate all the queue managers and applications to the latest version.

The side-by-side migration scenario is less flexible than multi-stage migration, and might not seem to have any advantages over it. However, side-by-side migration does have advantages over the multi-stage and single-stage approaches. With the side-by-side approach, because you uninstall the earlier version before starting any queue managers, you can assign an installation on the latest version to be the primary installation.

In the multi-stage approach, you cannot set an installation of the latest version to be the primary installation while you continue to run the earlier version.

Having the latest version installation as the primary installation has two benefits.

1. With the latest version having the primary installation, many applications restart without reconfiguring their environment.
2. IBM WebSphere MQ commands run against the primary installation, work without providing a local search path.

The advantage the side-by-side scenario has over the single-stage scenario is that you can install and verify the installation of the latest version of the product on the server before switching over to it.

The description of the migration scenario starts with the example in Figure 74 on page 403.

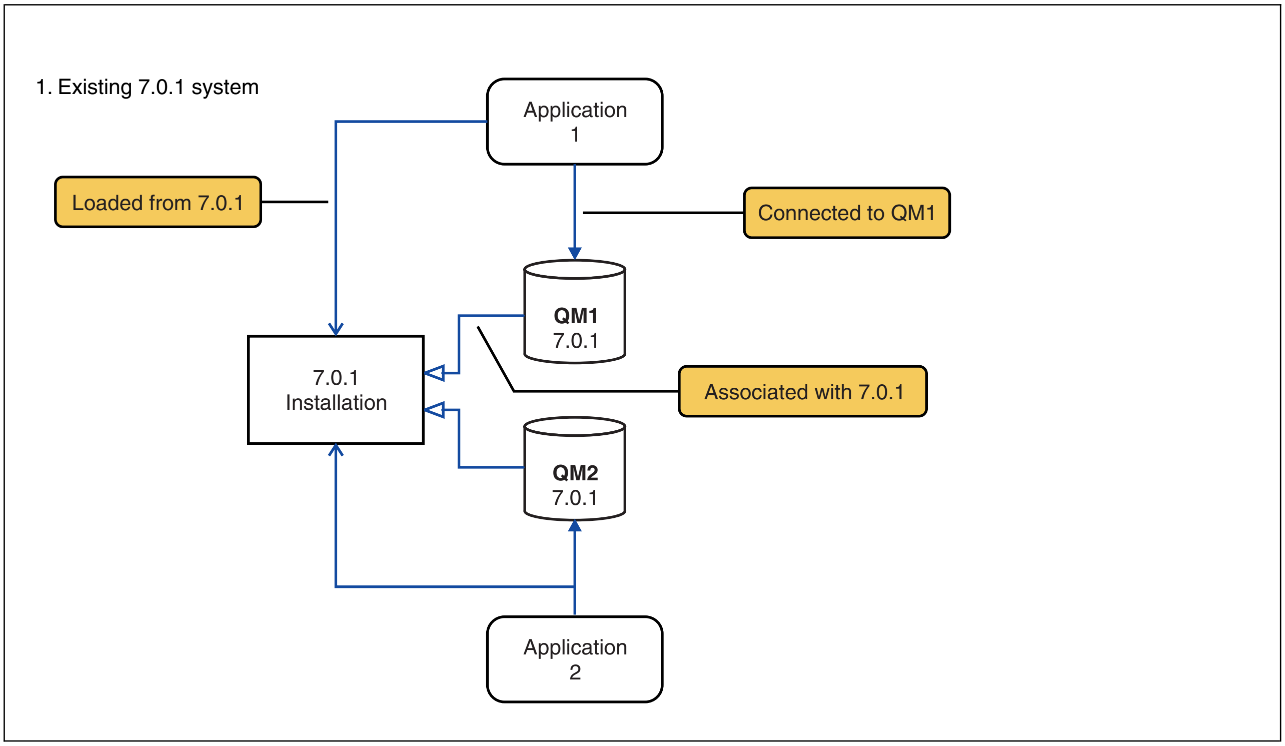


Figure 74. Existing Version 7.0.1 server

Four types of object are important to consider during migration: installations, queue managers, administrative procedures, and applications. The diagram shows the installation an application loads IBM WebSphere MQ libraries from, connections between applications and queue managers, and associations between queue managers and installations. Administrative procedures are omitted from the diagram. Administrative procedures contain IBM WebSphere MQ commands, and scripts that use commands.

“Loading from 7.0.1” in Figure 70 on page 396, refers to the IBM WebSphere MQ installation from which the application loads the IBM WebSphere MQ library it requires; see “Loading IBM WebSphere MQ libraries” on page 350. The connection is a call to MQCONN or MQCONNX, and has not changed from the earlier version of the product to the latest version. The association is the installation that the queue manager is associated with. The association is created either by running the **setmqm** command, or by starting a queue manager on the earlier version; see Associating a queue manager with an installation.

To run a command, the operating system must find the command in a IBM WebSphere MQ installation. For some commands, you must run the command from the installation that is associated with the correct queue manager. IBM WebSphere MQ does not switch commands to the correct installation. For other commands, such as **setmqinst**, you can run the command from any installation that has the latest version of the product installed.

If an earlier version of the product is installed, the command that is run is the command for that version, unless the search path is overridden by a local setting. You can override the search path by running **setmqenv**. If Version 7.0.1 is not installed, you must set the correct path to run a command. If you have set a primary installation, the command that is run is the copy in the primary installation, unless you override the selection with a local search path.

Procedure

1. Install the latest version in a different installation directory from the earlier version and verify the installation; see Figure 75 on page 404.

a. Decide on an installation naming convention. Give the installation a name of your choosing, or accept the default installation name. For the first installation, the default name is *Installation1*. For the second installation, the name is *Installation2*, and so on.

b. Verify the installation.

   Run the installation verification procedures and your own tests.

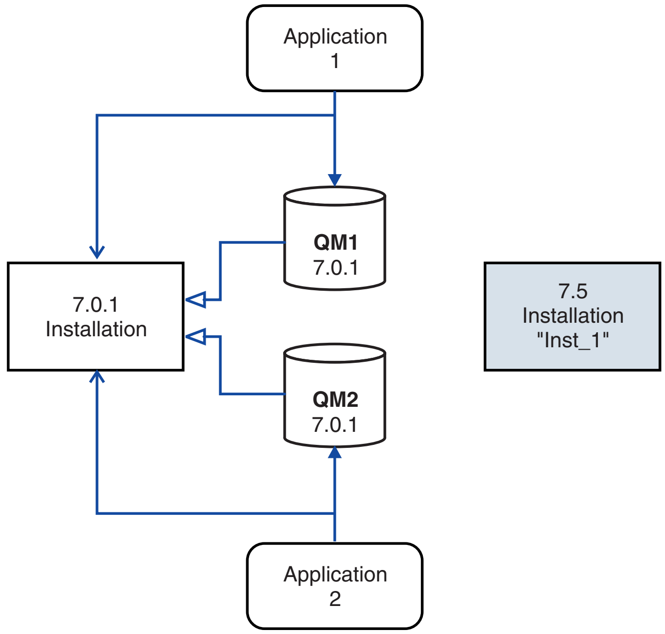


Figure 75. Install the latest version in a different directory

2. Uninstall the earlier version of the product; see Figure 76 on page 405.

   When uninstalling the earlier product, you must stop all queue managers and applications that have loaded a IBM WebSphere MQ library on the server. For this reason, you might choose to postpone uninstalling the earlier version of the product until a convenient maintenance window. When an earlier version of the product is not installed on a server, it is sufficient to stop the queue managers and applications that have loaded libraries from the installation that you are uninstalling or updating. It is not necessary to stop applications and queue managers associated with other installations.

   a. Stop all applications that have loaded IBM WebSphere MQ libraries on the server.

   b. Stop the queue managers and listeners on the server.

   c. Uninstall the earlier version of the product.

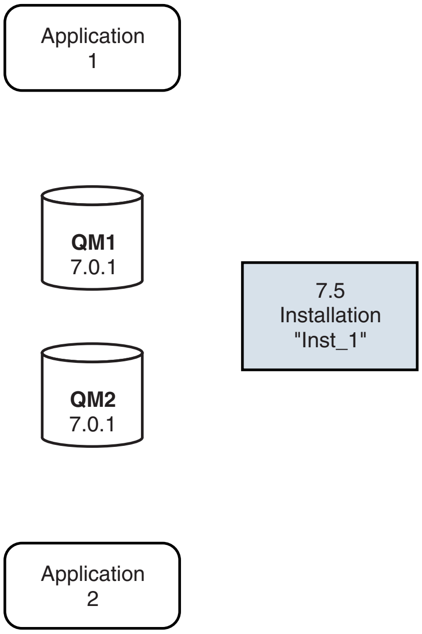


Figure 76. Uninstall an earlier version of the product

3. Make the latest version of the installation the primary installation; see Figure 77 on page 406.
   a. Run the **setmqinst** command

On Windows

```
"Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
```

On UNIX

```
Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
```

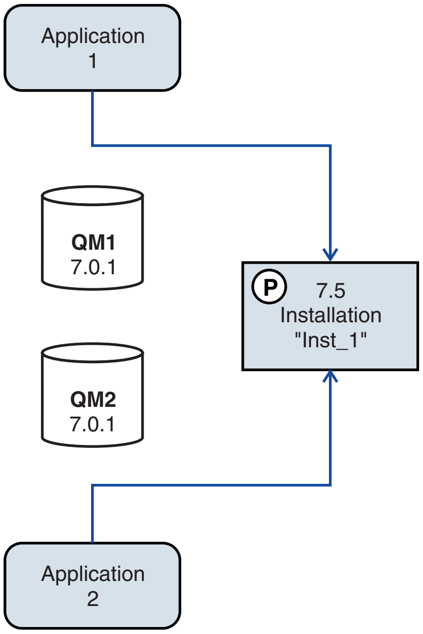


Figure 77. Prepare the installation and applications

- Make the installation primary to avoid specifying a search path to run IBM WebSphere MQ commands.
- If there is a primary installation, UNIX and Linux applications that expect to find the IBM WebSphere MQ library in `/usr/lib`, find a symbolic link to the library in `/usr/lib/32`[7]. `/usr/lib/32` is normally in the default search path. It is also specified as a load path in the IBM WebSphere MQ build scripts for UNIX and Linux.
- It is sufficient to link applications only to `/usr/lib`. With a primary installation of the latest version of the product defined on the server, an application can connect to any queue manager associated with any installation on the server. IBM WebSphere MQ loads the correct library for the application.

4. Start the queue managers and applications; see Figure 78 on page 407.
   a. Optional: Run the **setmqm** command to associate the queue managers with `Inst_1`.

```
setmqm -m QM1 -n Inst_1
setmqm -m QM2 -n Inst_1
```

   Running **setmqm** to associate the queue managers with `Inst_1` is optional, because these queue managers are being started on the latest version of the product for the first time.

   Windows If you have multiple installations, note that queue managers that were configured to start automatically, and remain after uninstalling IBM WebSphere MQ Version 7.0.1, automatically start under any other existing Version 7.1 (or laer) installation when either the machine reboots, or the Service for that installation is restarted. In order to avoid this, ensure that all queue managers have been moved to the desired installation before uninstalling IBM WebSphere MQ Version 7.0.1.

   b. Run the **strmqm** command to start the queue managers and migrate them to the latest version of the product.

[7] `/usr/lib` for 64 bit applications.

```
strmqm QM1
strmqm QM2
```

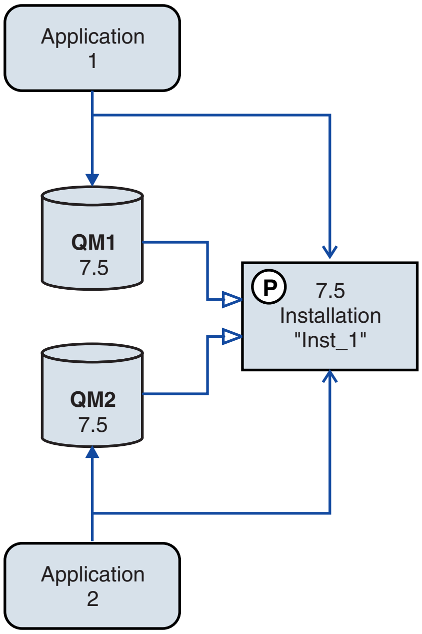


Figure 78. Restart the queue managers and applications

- When an application connects to a queue manager, the operating system searches its load path to load the IBM WebSphere MQ library[8]. A Version 7.5, or later, library contains code that checks that the queue manager is associated with an installation. If a queue manager is associated with a different installation, IBM WebSphere MQ loads the correct IBM WebSphere MQ library for the installation the queue manager is associated with.

What to do next

You cannot reinstall an earlier version of the product on a system that has the latest, or any other, version of IBM WebSphere MQ installed.

“Planning IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5 migration by platform” on page 374
Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

Installing WebSphere MQ server on AIX
You can install IBM WebSphere MQ server on an AIX system.

Uninstalling WebSphere MQ on AIX
On AIX, you can uninstall theIBM WebSphere MQ server or client using the System Management Interface Tool (SMIT) or the **installp** command.

Uninstalling WebSphere MQ on HP-UX
On HP-UX, you can uninstall theIBM WebSphere MQ server or client using the **swremove** command.

[8] On Windows, the IBM WebSphere MQ library is a DLL. A DLL is sometimes called a load library or a shared library. The entry points to a DLL are defined in a link library, with the file extension `.lib32` or `.lib`. The `.lib` library is linked at build-time and the DLL loaded at runtime.

Installing WebSphere MQ server on HP-UX

Uninstalling WebSphere MQ on Linux
On Linux, you can uninstall theIBM WebSphere MQ server or client using the **rpm** command.

Installing WebSphere MQ server on Linux
You can install a IBM WebSphere MQ server on either a 32-bit or 64-bit Linux system.

Uninstalling WebSphere MQ on Solaris
On Solaris, you can uninstall theIBM WebSphere MQ server or client using the **pkgrm** command.

Installing WebSphere MQ server on Solaris
You can install an IBM WebSphere MQ server on Solaris either interactively or silently.

Uninstalling WebSphere MQ on Windows systems
You can uninstall the IBM WebSphere MQ MQI clients and servers on Windows systems by using the control panel, the command line (**msiexec**), **MQParms**, or by using the installation media, in which case you can optionally remove queue managers as well.

Installing WebSphere MQ server on Windows
This topic describes how to install IBM WebSphere MQ server on Windows systems by using the Launchpad. This procedure can be used for installing a first or a subsequent installation.

Associating a queue manager with an installation
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must

modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

setmqenv

setmqinst

setmqm

UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5

Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

Before you begin

If you are using IBM WebSphere MQ Version 7.0.1, you must ensure that you are running IBM WebSphere MQ Version 7.0.1.6 before installing the latest version of the product on the same server. Go to Fix Central to obtain the fix pack.

This scenario is one of three, which describe alternative ways to upgrade queue managers from an earlier version of the product. The other scenarios are as follows:

1. Replace the earlier version with the latest version; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394.
2. Install the latest version of the product alongside an earlier version ; see "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401.

Read these three tasks to plan how you are going to migrate to the multi-installation environment of Version 7.5. The multi-stage migration scenario is the most flexible approach to migrating from Version 7.0.1 to Version 7.5.

These topics are for planning multi-installation migration. The planning topics guide you in deciding what other tasks you must perform to migrate queue managers and applications to the latest version. For the precise sequence of commands to upgrade a queue manager to the latest version, do the migration task for the platform you are interested in. All the tasks are listed by platform in the links at the end of this topic. As part of the queue manager migration task, back up your existing queue manager data. Even on a multi-installation server, queue managers cannot be restored to a previous command level after migration.

Note:

- If an application uses COM or ActiveX it can connect to any queue manager as long as there is a primary installation and it is Version 7.5 or later.
- If you are running the IBM WebSphere MQ .NET monitor in transactional mode, the queue manager it connects to must be the primary installation.

You cannot migrate these applications to Version 7.5 until you uninstall Version 7.0.1.

About this task

In the multi-stage migration scenario, you install IBM WebSphere MQ Version 7.5 alongside running queue managers that continue to be associated with Version 7.0.1. You can create queue managers and run new applications using the IBM WebSphere MQ Version 7.5 installation. When you are ready to start migrating queue managers and applications from Version 7.0.1, you can do so, one-by-one. When migration to Version 7.5 is complete, uninstall Version 7.0.1, and make the Version 7.5 installation primary.

With the multi-stage approach, until you uninstall version 7.0.1, you must configure an environment to run applications that connect to a version 7.1 queue manager. You must also provide a path to run IBM WebSphere MQ commands. Both these tasks are accomplished with the **setmqenv** command.

Note: When you have uninstalled Version 7.0.1, and set a Version 7.5 installation primary, in most circumstances it is not necessary to run the **setmqenv** command to run applications. It is still necessary to run **setmqenv** to set the environment for commands that connect to a queue manager associated with an installation that is not primary.

The description of the migration scenario starts with the example in Figure 79.

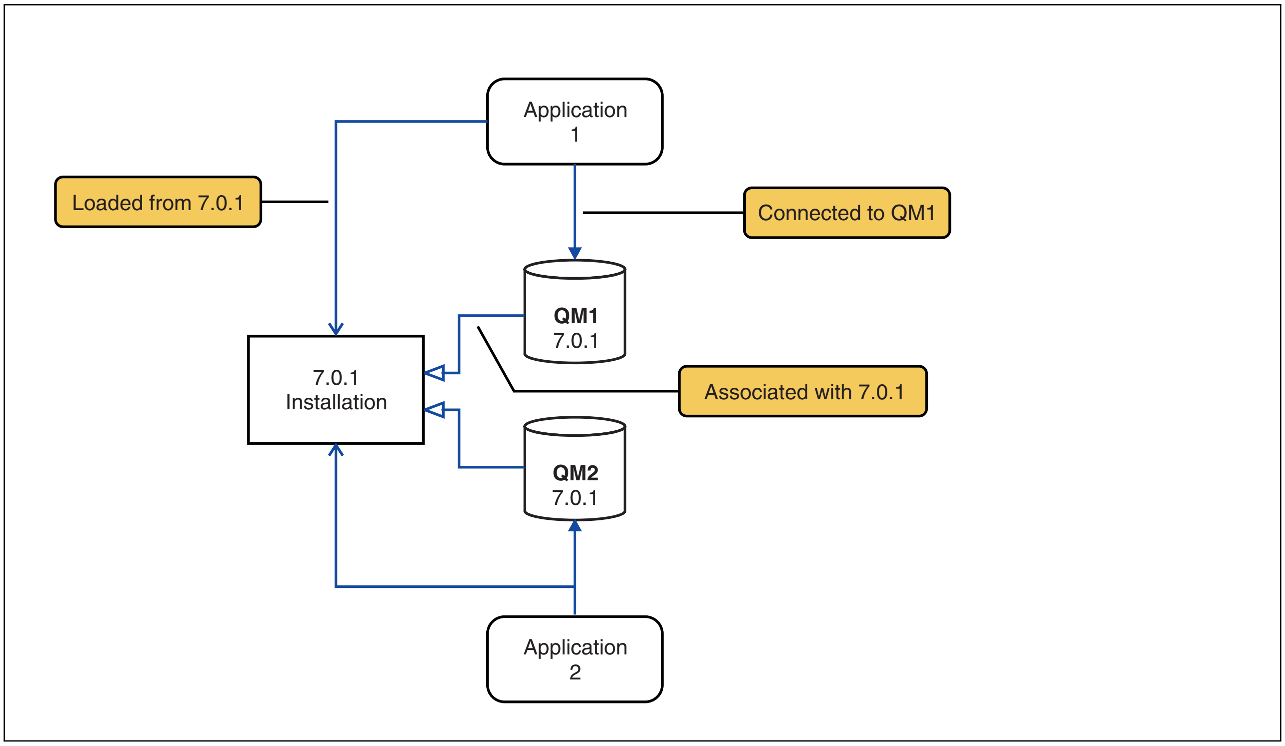


Figure 79. Existing Version 7.0.1 server

Four types of object are important to consider during migration: installations, queue managers, administrative procedures, and applications. The diagram shows the installation an application loads IBM WebSphere MQ libraries from, connections between applications and queue managers, and associations between queue managers and installations. Administrative procedures are omitted from the diagram. Administrative procedures contain IBM WebSphere MQ commands, and scripts that use commands.

“Loading from 7.0.1” in Figure 70 on page 396, refers to the IBM WebSphere MQ installation from which the application loads the IBM WebSphere MQ library it requires; see “Loading IBM WebSphere MQ libraries” on page 350. The connection is a call to MQCONN or MQCONNX, and has not changed from the earlier version of the product to the latest version. The association is the installation that the queue manager is associated with. The association is created either by running the **setmqm** command, or by starting a queue manager on the earlier version; see Associating a queue manager with an installation.

To run a command, the operating system must find the command in a IBM WebSphere MQ installation. For some commands, you must run the command from the installation that is associated with the correct queue manager. IBM WebSphere MQ does not switch commands to the correct installation. For other commands, such as **setmqinst**, you can run the command from any installation that has the latest version of the product installed.

If an earlier version of the product is installed, the command that is run is the command for that version, unless the search path is overridden by a local setting. You can override the search path by running **setmqenv**. If Version 7.0.1 is not installed, you must set the correct path to run a command. If you have set a primary installation, the command that is run is the copy in the primary installation, unless you override the selection with a local search path.

Procedure

1. Install Version 7.5 in a different installation directory to Version 7.0.1 and verify the installation; see Figure 80.
   a. Decide on an installation naming convention. Give the installation a name of your choosing, or accept the default installation name. For the first installation, the default name is *Installation1*. For the second installation, the name is *Installation2*, and so on.
   b. Verify the installation.

      Run the installation verification procedures and your own tests.

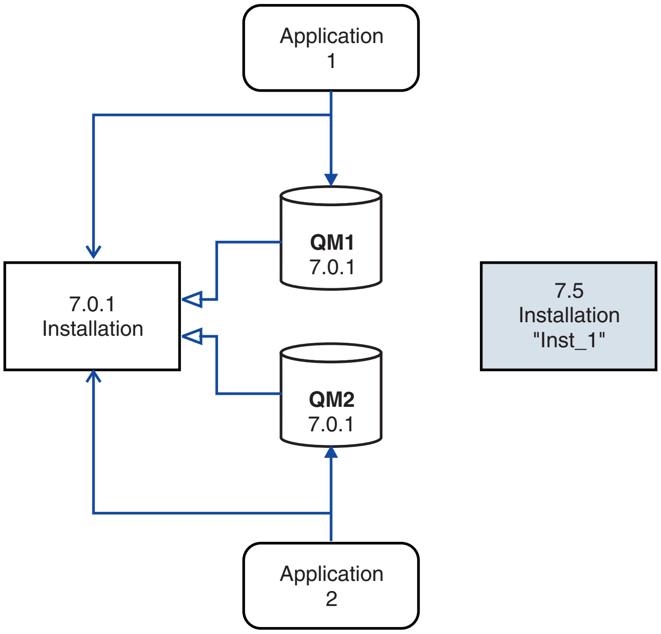


Figure 80. Install Version 7.5 in a different directory

   - You might create new queue managers running Version 7.5, and start to develop new applications before migrating applications from Version 7.0.1.
2. Configure the operating system so that applications load the Version 7.5 libraries; see Figure 81 on page 412.

Migrate queue managers one at a time. The first set of applications to load the Version 7.5 libraries are the applications that connect to the first queue manager you are going to migrate. It does not matter if those applications also connect to other queue managers on the server. If they load the Version 7.5 libraries, IBM WebSphere MQ automatically loads the Version 7.0.1 libraries for those applications that connect to Version 7.0.1. As the first step, you might either migrate the operating system environment of all applications, or just the applications that connect to the first queue manager you are going to migrate.

Some of the applications might be running as IBM WebSphere MQ MQI client applications on another workstation. When you migrate a queue manager, clients connected to it continue to run without loading a Version 7.5 client library. You can migrate these clients later, when you need to do so.

If any IBM WebSphere MQ MQI client applications are using the Version 7.0.1 library on the server, you must eventually migrate the clients to use Version 7.5 libraries, before you uninstall Version 7.0.1.

To make an application load a Version 7.5 library, you have three choices:

- Run **setmqenv** to modify the local path that is searched for IBM WebSphere MQ libraries.
- Modify the global search path that is searched for IBM WebSphere MQ libraries.
- Relink applications with an additional runtime load path.

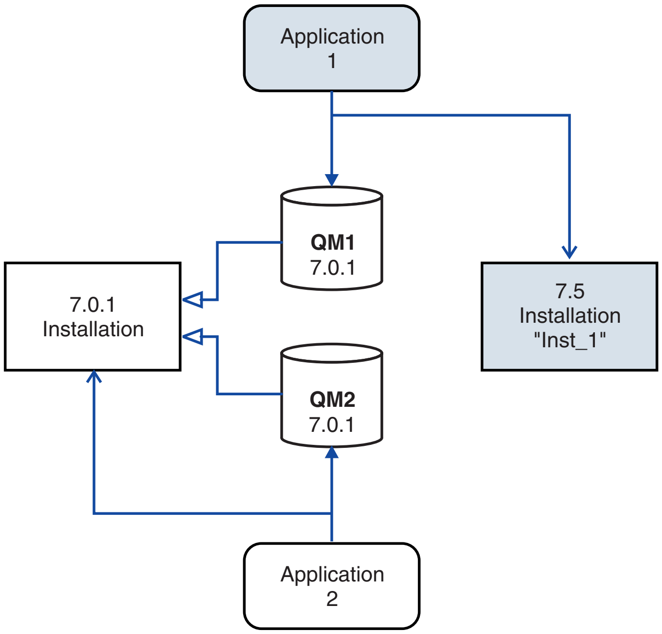


Figure 81. Application 1 loads IBM WebSphere MQ libraries from `Inst_1`

Consult operating system documentation about how to modify the global search path, or include a fixed runtime load path in the application load module.

To run **setmqenv** using the `-s` option:

Windows:

```
"Inst_1_INSTALLATION_PATH\bin\setmqenv" -s
```

The -s option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
. Inst_1_INSTALLATION_PATH/bin/setmqenv -s -k
```

The -k option inserts the path to the IBM WebSphere MQ load libraries at the start of the LD_LIBRARY_PATH environment variable, and adds the variable to the local environment; see "Loading IBM WebSphere MQ libraries" on page 350.

Note: On UNIX the leading ". " is critical. The dot followed by a space instructs the command shell run **setmqenv** in the same command shell and inherit the environment set by **setmqenv**.

3. Restart the queue manager and the applications that connect to it; see Figure 82.

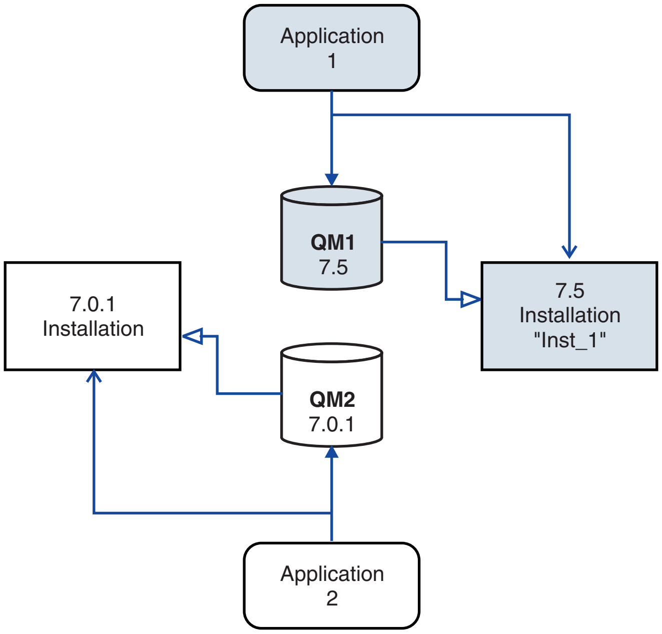


Figure 82. Restart QM1 and application 1.

a. Set up the local environment to the installation Inst_1.

Windows:

```
"Inst_1_INSTALLATION_PATH\bin\setmqenv" -s
```

The -s option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
. Inst_1_INSTALLATION_PATH/bin/setmqenv -s
```

b. Run the **setmqm** command to associate QM1 with Inst_1.

```
setmqm -m QM1 -n Inst_1
```

c. Run the **strmqm** command to start QM1 and migrate it to Version 7.5.

```
strmqm QM1
```

d. Restart application 1

The application loads the Version 7.5 library and connects to QM1, which is associated with Version 7.5.

4. Migrate all queue managers and applications to Version 7.5; see Figure 83.

Repeat steps 2 on page 411 and 3 on page 413, when required, until all the queue managers and applications are migrated to Version 7.5.

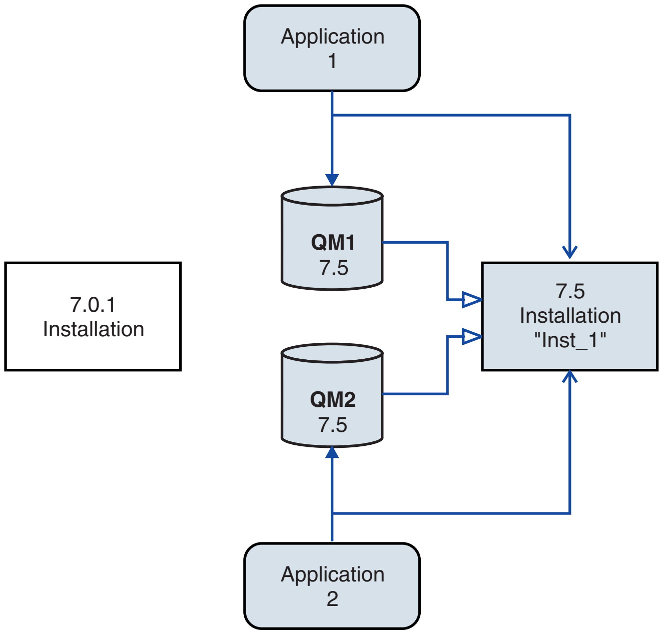


Figure 83. Migrate all queue managers and applications to Version 7.5

5. Uninstall Version 7.0.1; see Figure 84 on page 415.

When uninstalling the earlier product, you must stop all queue managers and applications that have loaded a IBM WebSphere MQ library on the server. For this reason, you might choose to postpone uninstalling the earlier version of the product until a convenient maintenance window. When an earlier version of the product is not installed on a server, it is sufficient to stop the queue managers and applications that have loaded libraries from the installation that you are uninstalling or updating. It is not necessary to stop applications and queue managers associated with other installations.

a. Stop all applications that have loaded IBM WebSphere MQ libraries on the server.

b. Stop the queue managers and listeners on the server.

c. Uninstall the earlier version of the product.

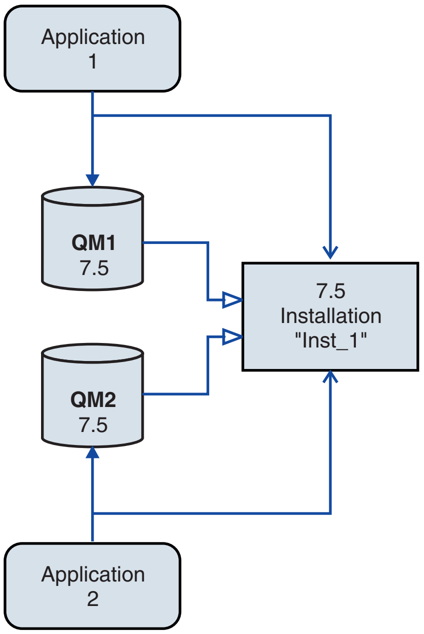


Figure 84. Uninstall Version 7.0.1

6. Make `Inst_1` the primary installation; see Figure 85 on page 416.
   a. Run the **setmqinst** command

On Windows

```
"Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
```

On UNIX

```
Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
```

- You do not have to set up a search path to run IBM WebSphere MQ commands from the primary installation.
- If you set a Version 7.5 installation as primary on UNIX and Linux, you do not have to set up `LD_LIBRARY_PATH` in most cases. You can remove calls to **setmqenv** to set `LD_LIBRARY_PATH`.

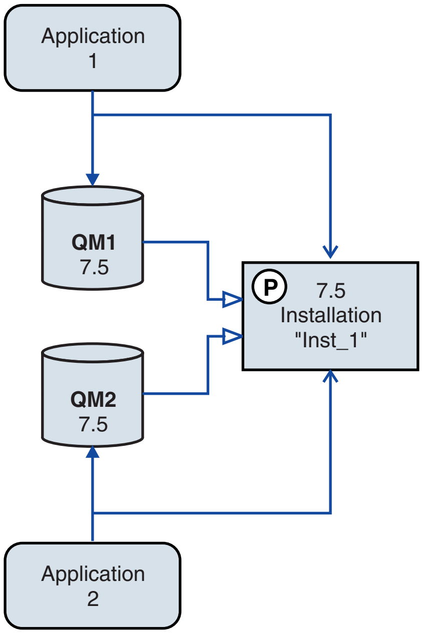


Figure 85. Make Version 7.5 primary

What to do next

You cannot reinstall an earlier version of the product on a system that has the latest, or any other, version of IBM WebSphere MQ installed.

Now that you have uninstalled Version 7.0.1, and made a Version 7.5 installation primary, you can review how the application runtime environment is set. It is no longer necessary to run **setmqenv** to set up the search path to load IBM WebSphere MQ libraries. If you have only one Version 7.5 installation it is not necessary to run **setmqenv** to run commands.

Related concepts:

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks:

"AIX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 384
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

"AIX: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 459
Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

"HP-UX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 386
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

"HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 463
Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

"Linux: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 388
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

"Linux: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 466
Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

"Solaris: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 390
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

"Solaris: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 470
Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

"Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 392
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

"Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 474
Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Related information:

Installing WebSphere MQ server on AIX
You can install IBM WebSphere MQ server on an AIX system.

Installing WebSphere MQ server on HP-UX

Installing WebSphere MQ server on Linux
You can install a IBM WebSphere MQ server on either a 32-bit or 64-bit Linux system.

Installing WebSphere MQ server on Solaris
You can install an IBM WebSphere MQ server on Solaris either interactively or silently.

Installing WebSphere MQ server on Windows
This topic describes how to install IBM WebSphere MQ server on Windows systems by using the Launchpad. This procedure can be used for installing a first or a subsequent installation.

Associating a queue manager with an installation
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

setmqenv

setmqinst

setmqm

Planning IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5 migration by platform

Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

If there are concepts about migration you do not understand, read "Introduction to WebSphere MQ migration" on page 320 first.

AIX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 6.0, see Table 52 on page 322 in “Where to find a topic about a specific migration path” on page 322.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 596.
3. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
4. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
5. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
6. Plan to migrate your queue manager to Version 7.5.

   See “AIX: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 493.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
7. Plan to update any manual or automated procedures you have written with changes to messages and codes.
8. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 596.

   Consider whether the application must be able to run on both Version 6.0 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
9. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

   Include the procedures and applications you identified in steps 6 and 7 in your regression tests.

10. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.
11. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Publish/Subscribe migration from version 6.0

"Migrating publish/subscribe from Version 6.0 to Version 7.5" on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

HP-UX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 6.0, see Table 52 on page 322

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
4. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
5. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrate a queue manager in a high availability configuration" on page 365.
6. Plan to migrate your queue manager to Version 7.5.

   See "HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 499.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
7. Plan to update any manual or automated procedures you have written with changes to messages and codes.
8. Plan to update applications that might be affected by changes you identified in "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.

   Consider whether the application must be able to run on both Version 6.0 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
9. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

   Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
10. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

11. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM WebSphere MQ migration” on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

“IBM WebSphere MQ upgrade” on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Publish/Subscribe migration from version 6.0

“Migrating publish/subscribe from Version 6.0 to Version 7.5” on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

Linux: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 6.0, see Table 52 on page 322

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 596.
3. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
4. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
5. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
6. Plan to migrate your queue manager to Version 7.5.

   See “Linux: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 503.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
7. Plan to update any manual or automated procedures you have written with changes to messages and codes.
8. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 596.

   Consider whether the application must be able to run on both Version 6.0 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
9. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

   Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
10. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

11. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Related information:

Changing configuration information on UNIX systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

➥ Fix Central

➥ IBM Passport Advantage

Publish/Subscribe migration from version 6.0

"Migrating publish/subscribe from Version 6.0 to Version 7.5" on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Solaris: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 6.0, see Table 52 on page 322

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
4. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
5. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrate a queue manager in a high availability configuration" on page 365.
6. Plan to migrate your queue manager to Version 7.5.

   See "Solaris: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 506.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
7. Plan to update any manual or automated procedures you have written with changes to messages and codes.
8. Plan to update applications that might be affected by changes you identified in "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.

   Consider whether the application must be able to run on both Version 6.0 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
9. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

   Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
10. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

11. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Related information:

Changing configuration information on UNIX systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Directory structure on UNIX and Linux systems
The WebSphere MQ directory structure on UNIX and Linux systems can be mapped to different file systems for easier management, better performance, and better reliability.

Fix Central

IBM Passport Advantage

Publish/Subscribe migration from version 6.0

"Migrating publish/subscribe from Version 6.0 to Version 7.5" on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Windows: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Before you begin

If the queue manager is currently not on Version 6.0, see Table 52 on page 322

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 596.
3. Review performance changes.

   Performance reports are published as Supportpacs; see WebSphere MQ - SupportPacs by Product.
4. Review the latest README file for the product you are working with.

   See WebSphere MQ and MQSeries product READMEs.
5. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see “Migrate a queue manager in a high availability configuration” on page 365.
6. Plan to migrate your queue manager to Version 7.5.

   See “Windows: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 509.

   Backing up queue manager data is part of the queue manager migration task.

   An alternative approach to backing up queue manager data, is to install and configure a new server. Test Version 7.5 with a new queue manager on the new server. When you are ready to go into production on Version 7.5 copy the queue manager configuration and data, to the new server.
7. Plan to update any manual or automated procedures you have written with changes to messages and codes.
8. Plan to update applications that might be affected by changes you identified in “Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5” on page 596.

   Consider whether the application must be able to run on both Version 6.0 and Version 7.5. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call MQINQ setting the `MQIA_COMMAND_LEVEL` selector.
9. Decide on what regression tests to perform before putting the queue manager into production on Version 7.5.

   Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
10. Plan to upgrade your WebSphere MQ MQI client installations to Version 7.5.

11. Plan to upgrade your client and server applications to use new functions in Version 7.5.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Directory structure on Windows systems
How to find queue manager configuration information and directories on Windows.

Fix Central

IBM Passport Advantage

“Migrate a queue manager in a high availability configuration” on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

“Queue manager cluster migration” on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

WebSphere MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS” on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM WebSphere MQ migration” on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

“IBM WebSphere MQ upgrade” on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Publish/Subscribe migration from version 6.0

“Migrating publish/subscribe from Version 6.0 to Version 7.5” on page 521
Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

Migration tasks

Follow these tasks to migrate queue managers from one version to another, and to migrate queue managers that are in clusters. You will also find assistance in migrating publish/subscribe or telemetry from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ.

Related concepts:

"Migration paths" on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

Related reference:

"List of changes by version, release, and maintenance level" on page 554
Any changes in the behavior of WebSphere MQ that affect existing applications or configurations are listed in the following topics. The changes are listed by migration path. For each migration path, all changes are listed. Study the changes to identify what migration tasks you must perform.

Migrating from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

The upgrade from Version 7.1 to Version 7.5 requires a full migration of queue managers. If you start a queue manager on Version 7.5, you can not restore it to Version 7.1.

Migrating a queue manager from Version 7.1 to Version 7.5

The procedures for migrating a queue manager from Version 7.1 to Version 7.5 are listed by platform.

Before you begin

If you have installed early support program code on the server, you must delete all the queue managers created with the installation. Uninstall the code before proceeding with installing the production level code.

About this task

Migration is a complex task. It goes beyond upgrading the IBM WebSphere MQ code. The term "upgrading" applies to the process of installing a new code level. "Migrating" refers to the process of upgrading code and the to the task of migrating queue manager data, such as messages, queues, channels, and other resources. Migrating queue manager data is automatic.

Procedure

Queue manager migration follows this overall plan:

1. Review the WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
2. Back up your system. In particular back up the queue manager.

   You cannot run a queue manager on a previous version of IBM WebSphere MQ, once you start the queue manager on the new version.
3. Carry out the platform-specific migration or upgrade procedure.

   Consult "Planning IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5 migration by platform" on page 374.
   a. Upgrade IBM WebSphere MQ.
   b. Customize the configuration.
   c. Verify the installation.

      Create a queue manager to verify the installation. Verify applications and management tasks work with the new level of IBM WebSphere MQ before migrating existing queue managers. Consult "Characteristics of different types of upgrade" on page 343 to help you decide on the extent of testing required.
4. Perform any additional migration tasks that are required.

If the queue manager is a member of a queue manager cluster, or queue-sharing group, migrate the other members of the cluster or group.

5. Perform any additional migration tasks that are required.

   If you are using publish/subscribe, you must migrate the publish/subscribe broker.

   If the queue manager is a member of a queue manager cluster, or queue-sharing group, migrate the other members of the cluster or group.

Related tasks:

"Planning IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5 migration by platform" on page 374
Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

"Migration paths" on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

IBM WebSphere MQ requirements

AIX: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "AIX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 374, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
4. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
5. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root.
6. Optionally, uninstall IBM WebSphere MQ Version 7.1. Note, that you carry out this step only if you are doing a single stage migration; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on AIX.
8. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See “UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version” on page 401 or “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409 for further information.
9. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   When you first start a queue manager after migration:

   - Any new attributes for existing objects are set to their default values.
   - Any new default objects are created.
   - Queue manager data is migrated.

Important: Do not use the -c option to start the queue manager, unless you explicitly want to reset or recreate the default system objects.

You must start IBM WebSphere MQ before you start any listeners.

What to do next

1. "AIX: Remedying the lack of shared storage to run 32 bit applications" on page 497.

   Backing up and restoring a queue manager
   Backing up queue managers and queue manager data.

   "The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
   On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

AIX: Restoring a Version 7.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
               ┌─-c─┐
►►──endmqm─────┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
               ├─-i─┤
               └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.

   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.

   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.

   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 7.1 queue manager.

What to do next

You might be restoring Version 7.1 on a server with multiple IBM WebSphere MQ installations. If one of the installations is primary, after restoring Version 7.1, the Version 7.1 installation will be by default, the primary. You must review how applications connect to an installation. After restoring Version 7.1 some applications might connect to the wrong installation.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “HP-UX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 376, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
4. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
5. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName─────────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root.
6. Optionally, uninstall IBM WebSphere MQ Version 7.1. Note, that you carry out this step only if you are doing a single stage migration; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on HP-UX.
8. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401 or "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409 for further information.
9. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   When you first start a queue manager after migration:
   - Any new attributes for existing objects are set to their default values.
   - Any new default objects are created.
   - Queue manager data is migrated.

   Important: Do not use the -c option to start the queue manager, unless you explicitly want to reset or recreate the default system objects.

   You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

“The version naming scheme for WebSphere MQ on platforms other than z/OS” on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

HP-UX: Restoring a Version 7.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌-c─┐
►►──endmqm──┼-w─┼──QmgrName──────────────────────────────────────────────►◄
            ├-i─┤
            └-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 7.1 queue manager.

What to do next

You might be restoring Version 7.1 on a server with multiple IBM WebSphere MQ installations. If one of the installations is primary, after restoring Version 7.1, the Version 7.1 installation will be by default, the primary. You must review how applications connect to an installation. After restoring Version 7.1 some applications might connect to the wrong installation.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Linux: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “Linux: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 378, as a guide.

2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. If you are considering switching IBM WebSphere MQ for Linux System x from a 32 bit to a 64 bit installation do not migrate to Version 7.5. Your last chance to switch to 64 bit is the migration to Version 7.0.1.
4. Do not use the **rpm** `-U` (upgrade) or `-F` (freshen) options to upgrade IBM WebSphere MQ for Linux; see "Linux: Cleaning up after using the **rpm** freshen or upgrade options" on page 439.
5. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
6. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
7. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                  ┌─-c─┐
      ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────►◄
                  ├─-i─┤
                  └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root.
6. Optionally, uninstall IBM WebSphere MQ Version 7.1. Note, that you carry out this step only if you are doing a single stage migration; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on Linux.
8. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See “UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version” on page 401 or “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409 for further information.
9. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   When you first start a queue manager after migration:

   - Any new attributes for existing objects are set to their default values.
   - Any new default objects are created.
   - Queue manager data is migrated.

   Important: Do not use the `-c` option to start the queue manager, unless you explicitly want to reset or recreate the default system objects.

   You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

“The version naming scheme for WebSphere MQ on platforms other than z/OS” on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Linux: Cleaning up after using the `rpm` freshen or upgrade options:

The use of **`rpm`** upgrade or freshen options is not supported. If you use the options, follow this cleanup procedure, and then install following the correct steps.

Before you begin

You have attempted to upgrade IBM WebSphere MQ for Linux using `rpm -U` or `rpm -F`

About this task

By using the freshen or upgrade options, you might have deleted your old IBM WebSphere MQ package entries from the **rpm** database without removing the product from your system. You might also have partially installed IBM WebSphere MQ

Procedure

Follow these steps to clean up your system.

1. Find out which IBM WebSphere MQ MQ packages still have entries in your RPM database.

   ```
   rpm -qa | grep MQSeries
   ```
2. Remove all remaining IBM WebSphere MQ packages from your system.

   ```
   rpm -e package-name
   ```
3. Remove the /opt/mqm directory.

   ```
   rm -rf /opt/mqm
   ```

Linux: Restoring a Version 7.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 7.1 queue manager.

What to do next

You might be restoring Version 7.1 on a server with multiple IBM WebSphere MQ installations. If one of the installations is primary, after restoring Version 7.1, the Version 7.1 installation will be by default, the primary. You must review how applications connect to an installation. After restoring Version 7.1 some applications might connect to the wrong installation.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Solaris: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “Solaris: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 380, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
4. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
5. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **`MQSC`** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **`endmqm`** command to stop each running queue manager associated with this installation.

```
          ┌─-c─┐
►►─endmqm─┼─-w─┼─QmgrName──────────────────────────────────────────────►◄
          ├─-i─┤
          └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root.
6. Optionally, uninstall IBM WebSphere MQ Version 7.1. Note, that you carry out this step only if you are doing a single stage migration; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on Solaris.
8. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401 or "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409 for further information.
9. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   When you first start a queue manager after migration:

   - Any new attributes for existing objects are set to their default values.
   - Any new default objects are created.
   - Queue manager data is migrated.

   Important: Do not use the `-c` option to start the queue manager, unless you explicitly want to reset or recreate the default system objects.

   You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

“The version naming scheme for WebSphere MQ on platforms other than z/OS” on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Solaris: Restoring a Version 7.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
               ┌-c┐
►►──endmqm─────┼-w┼──QmgrName──────────────────────────────────────────────────►◄
               ├-i┤
               └-p┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 7.1 queue manager.

What to do next

You might be restoring Version 7.1 on a server with multiple IBM WebSphere MQ installations. If one of the installations is primary, after restoring Version 7.1, the Version 7.1 installation will be by default, the primary. You must review how applications connect to an installation. After restoring Version 7.1 some applications might connect to the wrong installation.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "Windows: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 382, as a guide.

2. The migration paths to IBM WebSphere MQ Version 7.5 on Windows Vista are listed in Table 62 on page 510

Table 58. IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008: supported migration paths

From:	To:
IBM WebSphere MQ Version 6.0 plus fix pack 6.0.2.1 on Windows Vista or Windows Server 2008	IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008
IBM WebSphere MQ Version 7.0.1 on Windows XP Professional SP2	IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008

3. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
4. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
5. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
6. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.1.

All the objects that you previously created are maintained. The components that were previously installed are preselected in the feature options when you install the new level. If you leave these components selected, you can keep them or reinstall them. If you clear any of these components, the installation process uninstalls them. By default, a typical migration installs only the same features that were installed in the previous version installation.

For example, if IBM WebSphere MQ Explorer was not installed in the Version 7.1 installation, it is not installed in a Version 7.5 installation. If you want IBM WebSphere MQ Explorer, select a custom installation, and select the IBM WebSphere MQ Explorer feature on the Features panel. If you do not want IBM WebSphere MQ Explorer, uninstall the IBM WebSphere MQ Explorer feature by selecting a custom installation. Then clear the IBM WebSphere MQ Explorer feature on the Features panel. For more information about how to uninstall features, see Modifying the installation using WebSphere MQ Installation Launchpad.

You can also migrate a Version 7.1 queue manager to IBM WebSphere MQ Version 7.5 on a system where a previous version has been uninstalled. The queue manager data must have been retained, or restored from a backup.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Stop the MQSeries Service and exit the Service taskbar icon application.
6. Upgrade the IBM WebSphere MQ installation to Version 7.5 using one of these procedures:
   - Modifying the installation using WebSphere MQ Installation Launchpad
   - Silently modifying a WebSphere MQ server installation using **msiexec**
7. Reenter domain, user ID, and password information

   When the installation of IBM WebSphere MQ Version 7.5 completes, the Prepare WebSphere MQ Wizard starts automatically.

   On Windows Vista or Windows Server 2008 where UAC is enabled: If you rerun the Prepare WebSphere MQ Wizard, ensure that the wizard is run with Administrator privilege, otherwise the wizard might fail.
8. Optionally, uninstall IBM WebSphere MQ Version 7.1. Note, that you carry out this step only if you are doing a single stage migration; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394

What to do next

You might be restoring Version 7.1 on a server with multiple IBM WebSphere MQ installations. If one of the installations is primary, after restoring Version 7.1, the Version 7.1 installation will be by default, the primary. You must review how applications connect to an installation. After restoring Version 7.1 some applications might connect to the wrong installation.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Windows: Restoring a Version 7.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
                ┌-c-┐
►►──endmqm──────┼-w-┼──QmgrName────────────────────────────────────────────►◄
                ├-i-┤
                └-p-┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.

   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.

   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.

   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 7.1 queue manager.

What to do next

You might be restoring Version 7.1 on a server with multiple IBM WebSphere MQ installations. If one of the installations is primary, after restoring Version 7.1, the Version 7.1 installation will be by default, the primary. You must review how applications connect to an installation. After restoring Version 7.1 some applications might connect to the wrong installation.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Migrating a IBM WebSphere MQ MQI client from Version 7.1 to Version 7.5

Migrate a IBM WebSphere MQ MQI client from Version 7.1 to Version 7.5 by doing the tasks in the following topics. Check IBM WebSphere MQ MQI client applications with Version 7.5 before migration. You must stop all IBM WebSphere MQ activity on the workstation, before upgrading the IBM WebSphere MQ MQI client. After upgrading the IBM WebSphere MQ MQI client, you must check the client channel configuration.

AIX: Migrating a IBM WebSphere MQ MQI client for AIX from Version 7.0.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "AIX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 374, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for AIX installation on a workstation; see Client installation procedure on an AIX workstation.
   - To upgrade a IBM WebSphere MQ MQI client for AIX installation on an AIX IBM WebSphere MQ server; see Installing a IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"AIX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 374
Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Related information:

Client installation procedure on an AIX workstation
You can interactively install the IBM WebSphere MQ client for AIX using **smit**.

Installing WebSphere MQ MQI clients on the same machine as the server

HP-UX: Migrating a IBM WebSphere MQ MQI client for HP-UX from Version 7.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "HP-UX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 376, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for HP-UX installation on a workstation; see Client installation procedure on an HP-UX workstation.
   - To upgrade a IBM WebSphere MQ MQI client for HP-UX installation on an HP-UX IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"HP-UX: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 376
Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Related information:

Client installation procedure on an HP-UX workstation

Linux: Migrating a IBM WebSphere MQ MQI client for Linux from Version 7.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Linux: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 378, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

See changes from “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.

3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Linux installation on a workstation; see Client installation procedure on an Linux workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Linux installation on an Linux IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

“Linux: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 378
Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Related information:

Client installation procedure on a Linux workstation
Installing a WebSphere MQ client on a 32 bit or 64 bit Linux system.

Solaris: Migrating a IBM WebSphere MQ MQI client for Solaris from Version 7.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, “Solaris: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 380, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from “Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5” on page 558.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Solaris installation on a workstation; see Client installation procedure on an Solaris workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Solaris installation on an Solaris IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Solaris: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 380
Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Related information:

Client installation procedure on a Solaris workstation

Windows: Migrating a IBM WebSphere MQ MQI client for Windows from Version 7.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Windows: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 382, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 558.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Windows installation on a workstation; see Client installation procedure on an Windows workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Windows installation on an Windows IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Windows: Planning for migration from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 382
Create a migration plan to upgrade from Version 7.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.1 once you start it on Version 7.5.

Related information:

Client installation procedure on a Windcws workstation
This topic describes how to install IBM WebSphere MQ client on Windows systems. This procedure can be used for installing a first or a subsequent installation.

Restoring a Version 7.5 IBM WebSphere MQ MQI client and client connection to Version 7.1:

If you restore a Version 7.5 IBM WebSphere MQ MQI client to the Version 7.1 code level, you must undo the configuration changes manually.

About this task

It is unusual to restore earlier IBM WebSphere MQ MQI client libraries to a workstation. The principal tasks are listed in the following steps.

Procedure

1. End all IBM WebSphere MQ activity on the workstation.
2. Uninstall the Version 7.5 IBM WebSphere MQ MQI client code.
3. Follow the client installation procedure for the platform to install the Version 7.1IBM WebSphere MQ MQI client.
4. If you configured a Client Connection Definition Table (CCDT) from a Version 7.5 queue manager, revert to using a table created by a Version 7.1 queue manager.

   The CCDT must always be created by a queue manager on the same, or earlier, release to the client.

Migrating applications from Version 7.1 to Version 7.5

IBM WebSphere MQ applications do not require migration between Version 7.1 to Version 7.5.

Linux: Rebuilding a C++ application:

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.

Before you begin

1. Check the required level of GCC for your distribution of Linux; see IBM WebSphere MQ requirements.
2. If you are using SSL or TLS, also check the required level of `libstdc++.so`.
3. Check whether the application requires rebuilding. Run the following command to display what version of `libstdc++.so` the application depends upon. If the result is less than `libstdc++.so.6`, you must rebuild your application.

   ```
   ldd ApplicationPath
   ```

About this task

The task describes the steps required to rebuild a Linux C++ IBM WebSphere MQ application. For more detailed instructions about building Linux applications for IBM WebSphere MQ; see Building your application on Linux

Procedure

1. Check that the required GCC library is installed correctly.

   Run one of the following commands:

   - Check the 32 bit library on an x86 Linux system:

     ```
     ls -l /usr/lib/libstdc++.so.6
     ```

   - Check the 64 bit library on any other Linux system.

     ```
     ls -l /usr/lib64/libstdc++.so.6
     ```

2. Check that the GCC compiler is at least at version 4.1.2

   Run the following command to display the version of GCC.

   ```
   gcc -v
   ```
3. Rebuild the application

   The commands to compile and link Linux C++ applications are described in Building 32-bit applications and Building 64-bit applications

What to do next

When you deploy your Linux C++ application, ensure that the same GCC runtime library is correctly installed on the run time system.

Migrating from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

The upgrade from Version 7.0.1 to Version 7.5 requires a full migration of queue managers. On platforms other than z/OS, if you start a queue manager on Version 7.5, you can not restore it to Version 7.0.1.

Migrating IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5

Migrate IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5 by completing the tasks in this section. You must stop all IBM WebSphere MQ activity on the system before migrating.

In IBM WebSphere MQ Version 7.0.1, IBM WebSphere MQ Telemetry was a separate feature. Because IBM WebSphere MQ Telemetry is a component of IBM WebSphere MQ Version 7.5 and later, upgrading is essentially uninstalling IBM WebSphere MQ TelemetryVersion 7.0.1 and installing IBM WebSphere MQ Version 7.5. IBM WebSphere MQ Telemetry can either be installed with the main product, or installed after Version 7.5 or later is already installed.

After the successful upgrade, Linux systems retain all telemetry data kept in `/var/mqm`, and the Windows systems retain the telemetry data in the IBM WebSphere MQ Version 7.5 installation directory, for example: `C:\Program Files (x86)\IBM\WebSphere MQ\`. Telemetry data is migrated to Version 7.5 when the queue manager is started again.

Note:

You can administer IBM WebSphere MQ Telemetry Version 7.0.1 only from the Version 7.0.1 IBM WebSphere MQ Explorer. If you connect the Version 7.5 explorer remotely to a Version 7.0.1 queue manager, no telemetry resources are displayed. You cannot connect a Version 7.5 IBM WebSphere MQ Explorer locally to a Version 7.0.1 queue manager on the same server.

The migration tasks, depending on your platform, are described in the following subtopics:

Related concepts:

"IBM WebSphere MQ Telemetry migration from Version 7.0.1" on page 372
IBM WebSphere MQ Telemetry Version 7.0.1 is a separate feature that you install with IBM WebSphere MQ Version 7.0.1. Telemetry is integrated into IBM WebSphere MQ Version 7.5.

Related information:

Installing WebSphere MQ
The topics in this section provide instructions on how to install IBM WebSphere MQ.

Installing WebSphere MQ Telemetry
From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

Windows: Migrating IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5:

Follow these instructions to migrate IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5 on Windows.

Before you begin

Before proceeding with this task, ensure that you back up your existing IBM WebSphere MQ installation. You must stop the IBM WebSphere MQ Telemetry service `SYSTEM.MQXR.SERVICE` before migrating.

About this task

This task outlines the steps necessary to migrate your existing installation of IBM WebSphere MQ Telemetry to Version 7.5 on Windows systems.

Procedure

1. Uninstall IBM WebSphere MQ Version 7.0.1 using the control panel, following this procedure:
   a. From the Windows task bar, click **Start > Control Panel**. (On some systems, click **Start > Settings > Control Panel**)
   b. When uninstalling IBM WebSphere MQ Telemetry by using the control panel, on Windows 7 systems, click **Programs and Features**, on other Windows systems, click **Add or Remove Programs**.
   c. Click IBM WebSphere MQ Telemetry, then click **Change/Remove**. The uninstaller starts and summarizes items to be uninstalled.
   d. Click **Uninstall**. The uninstaller lists the items being uninstalled. The Uninstall Complete screen displays the status of the uninstall. Click **Done**
2. Verify that the data folders still exist. For example, they might be located at `C:\Program Files (x86)\IBM\WebSphere MQ\`.
3. IBM WebSphere MQ Telemetry is installed in one of two ways:
   - Installed as part of a **Custom** installation at the same time as IBM WebSphere MQ Version 7.0.1 during migration: Begin here: "Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 392
   - Added at a later date to an existing installation of IBM WebSphere MQ Version 7.5 or later: Begin here: Installing IBM WebSphere MQ Telemetry
4. Verify that the IBM WebSphere MQ Telemetry migration was successful by following the tasks at Verifying the installation of IBM WebSphere MQ Telemetry by using IBM WebSphere MQ Explorer or Verifying the installation of IBM WebSphere MQ Telemetry using the command line.

Results

After completing this task, the existing MQTT channels are still present. Previous subscriptions are also still present. System output similar to the following example is expected:

AMQ4616
A newer command level has been found when connecting to <*insert_0*>. The old level is <*insert_1*> and the new level is <*insert_2*>. The connection to the queue manager will be replaced.

Severity
0: Information

Explanation
A previous connection to this queue manager has been successful; the queue manager is the same but the command level is now higher. The version of WebSphere MQ has been changed.

Response

Message for information only.

Related information:

Installing WebSphere MQ Telemetry
From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

Verifying the installation of WebSphere MQ Telemetry
There are two ways to verify the installation of IBM WebSphere MQ Telemetry. Either one or both can be used, regardless of whether IBM WebSphere MQ Telemetry was installed as a custom installation of IBM WebSphere MQ, or added to an existing installation of IBM WebSphere MQ.

Verifying the installation of WebSphere MQ Telemetry by using WebSphere MQ Explorer
Use the Define sample configuration wizard and the MQTT client utility in IBM WebSphere MQ Explorer to verify that the IBM WebSphere MQ Telemetry components have installed. Also check that publish/subscribe works correctly.

Linux: Migrating from IBM WebSphere MQ Telemetry Version 7.0.1 to Version 7.5:

Follow these instructions to migrate IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5 on Linux.

Before you begin

Before proceeding with this task, ensure that you back up your existing IBM WebSphere MQ installation. You must stop the IBM WebSphere MQ Telemetry service `SYSTEM.MQXR.SERVICE` before migrating.

About this task

This task outlines the steps necessary to migrate your existing installation of IBM WebSphere MQ Telemetry to version Version 7.5 on Linux systems.

Procedure

1. Uninstall IBM WebSphere MQ Telemetry Version 7.0.1, following this procedure:
   a. Navigate to the uninstallation directory. The default location on Linux is `/opt/mqm/mqxr/Uninstall_MQTT`.
   b. Start the uninstaller, using the executable or binary file. On Linux systems, run `./Uninstall_MQTelemetry -i GUI`. The uninstaller starts and summarizes what is to be uninstalled.
   c. Click **Uninstall**. The uninstaller lists the items being uninstalled.
   d. Click **Done**.
2. Verify that the data folders still exist. Consult your *MQ_INSTALLATION_PATH* environment variable to find these folders.
3. IBM WebSphere MQ Telemetry is installed in one of two ways:
   - Installed as part of a **Custom** installation at the same time as IBM WebSphere MQ Version 7.0.1 during migration: Begin here: “Linux: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 388
   - Added at a later date to an existing installation of IBM WebSphere MQ Version 7.5 or later: Begin here: Installing IBM WebSphere MQ Telemetry
4. Verify that the IBM WebSphere MQ Telemetry migration was successful by following the tasks at Verifying the installation of IBM WebSphere MQ Telemetry by using IBM WebSphere MQ Explorer or Verifying the installation of IBM WebSphere MQ Telemetry using the command line.

Results

After completing this task, the existing MQTT channels are still present. Previous subscriptions are also still present. System output similar to the following example is expected:

AMQ4616
A newer command level has been found when connecting to <*insert_0*>. The old level is <*insert_1*> and the new level is <*insert_2*>. The connection to the queue manager will be replaced.

Severity
0: Information

Explanation
A previous connection to this queue manager has been successful; the queue manager is the same but the command level is now higher. The version of WebSphere MQ has been changed.

Response
Message for information only.

Related information:

Installing WebSphere MQ Telemetry
From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

Verifying the installation of WebSphere MQ Telemetry
There are two ways to verify the installation of IBM WebSphere MQ Telemetry. Either one or both can be used, regardless of whether IBM WebSphere MQ Telemetry was installed as a custom installation of IBM WebSphere MQ, or added to an existing installation of IBM WebSphere MQ.

Verifying the installation of WebSphere MQ Telemetry by using WebSphere MQ Explorer
Use the Define sample configuration wizard and the MQTT client utility in IBM WebSphere MQ Explorer to verify that the IBM WebSphere MQ Telemetry components have installed. Also check that publish/subscribe works correctly.

Migrating a queue manager from Version 7.0.1 to Version 7.5

The procedures for migrating a queue manager from Version 7.0.1 to Version 7.5 are listed by platform.

Before you begin

If you have installed early support program code on the server, you must delete all the queue managers created with the installation. Uninstall the code before proceeding with installing the production level code.

About this task

Migration is a complex task. It goes beyond upgrading the IBM WebSphere MQ code. The term “upgrading” applies to the process of installing a new code level. “Migrating” refers to the process of upgrading code and the to the task of migrating queue manager data, such as messages, queues, channels, and other resources. Migrating queue manager data is automatic.

Procedure

Queue manager migration follows this overall plan:

1. Review the WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
2. Back up your system. In particular back up the queue manager.

You cannot run a queue manager on a previous version of IBM WebSphere MQ, once you start the queue manager on the new version.

3. Carry out the platform-specific migration or upgrade procedure.

   Consult “Planning IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 migration by platform” on page 384.

   a. Upgrade IBM WebSphere MQ.
   b. Customize the configuration.
   c. Verify the installation.

      Create a queue manager to verify the installation. Verify applications and management tasks work with the new level of IBM WebSphere MQ before migrating existing queue managers. Consult “Characteristics of different types of upgrade” on page 343 to help you decide on the extent of testing required.

4. Perform any additional migration tasks that are required.

   If the queue manager is a member of a queue manager cluster, or queue-sharing group, migrate the other members of the cluster or group.

5. Perform any additional migration tasks that are required.

   If you are using publish/subscribe, you must migrate the publish/subscribe broker.

   If the queue manager is a member of a queue manager cluster, or queue-sharing group, migrate the other members of the cluster or group.

Related tasks:

“Planning IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 migration by platform” on page 384
Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

“Migration paths” on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

IBM WebSphere MQ requirements

AIX: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “AIX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 384, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.0.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
4. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
5. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.0.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   1. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   2. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   3. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                    ┌─-c─┐
      ►►──endmqm────┼─-w─┼──QmgrName──────────────────────────────────────►◄
                    ├─-i─┤
                    └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.
   4. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Back up the queue manager.
5. Log in as root.

6. If you are running IBM WebSphere MQ Version 7.0.1.6 or later, optionally uninstall the current version of IBM WebSphere MQ. Note, that you carry out this step only if you are doing a single stage migration; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394
7. If you are running IBM WebSphere MQ Version 7.0.1.5 or earlier, uninstall the current version of IBM WebSphere MQ. If you require a side-by-side migration or multistage migration only, you must upgrade to IBM WebSphere MQ Version 7.0.1.6
8. Install IBM WebSphere MQ Version 7.5.

   See Installing WebSphere MQ Server on AIX.
9. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See “UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version” on page 401 or “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409 for further information.
10. Start the queue manager.

    ```
    strmqm QmgrName
    ```

    When you first start a queue manager after migration:
    - Any new attributes for existing objects are set to their default values.
    - Any new default objects are created.
    - Queue manager data is migrated.

    Important: Do not use the `-c` option to start the queue manager, unless you explicitly want to reset or recreate the default system objects.

    You must start IBM WebSphere MQ before you start any listeners.

What to do next

1. “AIX: Remedying the lack of shared storage to run 32 bit applications” on page 497.

   Backing up and restoring a queue manager
   Backing up queue managers and queue manager data.

   “The version naming scheme for WebSphere MQ on platforms other than z/OS” on page 324
   On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

AIX: Restoring a Version 7.0.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.0.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.0.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.

3. You cannot restore a Version 7.0.1 installation to a server that has multiple IBM WebSphere MQ installations.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                  ┌─-c─┐
      ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────►◄
                  ├─-i─┤
                  └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.
   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Restore the system, or IBM WebSphere MQ and the queue manager.

If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 7.0.1 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “HP-UX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 386, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.0.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
4. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
5. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.0.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.

   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

   ```
              ┌─-c─┐
   ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
              ├─-i─┤
              └─-p─┘
   ```

   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

   You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

   Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

   ```
   endmqlsr -m QMgrName
   ```

4. Back up the queue manager.
5. Log in as root.
6. If you are running IBM WebSphere MQ Version 7.0.1.6 or later, optionally uninstall the current version of IBM WebSphere MQ. Note, that you carry out this step only if you are doing a single stage migration; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394
7. If you are running IBM WebSphere MQ Version 7.0.1.5 or earlier, uninstall the current version of IBM WebSphere MQ. If you require a side-by-side migration or multistage migration only, you must upgrade to IBM WebSphere MQ Version 7.0.1.6
8. Install IBM WebSphere MQ Version 7.5.

   See Installing WebSphere MQ Server on HP-UX.
9. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See “UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version” on page 401 or “UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5” on page 409 for further information.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

HP-UX: Restoring a Version 7.0.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.0.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.0.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. You cannot restore a Version 7.0.1 installation to a server that has multiple IBM WebSphere MQ installations.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
              ┌─-c─┐
▸▸──endmqm────┼─-w─┼──QmgrName──────────────────────────────────────────▸◂
              ├─-i─┤
              └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 7.0.1 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Linux: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “Linux: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 388, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. You must uninstall IBM WebSphere MQ Telemetry before installing IBM WebSphere MQ Version 7.5; see “Migrating IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5” on page 455.
4. If you are considering switching IBM WebSphere MQ for Linux System x from a 32 bit to a 64 bit installation do not migrate to Version 7.5. Your last chance to switch to 64 bit is the migration to Version 7.0.1.

5. Do not use the **rpm** -U (upgrade) or -F (freshen) options to upgrade IBM WebSphere MQ for Linux; see “Linux: Cleaning up after using the **rpm** freshen or upgrade options” on page 439.
6. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.0.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
7. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
8. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.0.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
              ┌─-c─┐
      ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
              ├─-i─┤
              └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```

4. Back up the queue manager.
5. Log in as root.
6. If you are running IBM WebSphere MQ Version 7.0.1.6 or later, optionally uninstall the current version of IBM WebSphere MQ. Note, that you carry out this step only if you are doing a single stage migration; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
7. If you are running IBM WebSphere MQ Version 7.0.1.5 or earlier, uninstall the current version of IBM WebSphere MQ. If you require a side-by-side migration or multistage migration only, you must upgrade to IBM WebSphere MQ Version 7.0.1.6
8. Install IBM WebSphere MQ Version 7.5.

   See Installing WebSphere MQ Server on Linux.
9. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401 or "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409 for further information.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Linux: Cleaning up after using the `rpm` freshen or upgrade options:

The use of **`rpm`** upgrade or freshen options is not supported. If you use the options, follow this cleanup procedure, and then install following the correct steps.

Before you begin

You have attempted to upgrade IBM WebSphere MQ for Linux using `rpm -U` or `rpm -F`

About this task

By using the freshen or upgrade options, you might have deleted your old IBM WebSphere MQ package entries from the **`rpm`** database without removing the product from your system. You might also have partially installed IBM WebSphere MQ

Procedure

Follow these steps to clean up your system.

1. Find out which IBM WebSphere MQ MQ packages still have entries in your RPM database.

   ```
   rpm -qa | grep MQSeries
   ```

2. Remove all remaining IBM WebSphere MQ packages from your system.

```
rpm -e package-name
```

3. Remove the /opt/mqm directory.

   ```
   rm -rf /opt/mqm
   ```

Linux: Restoring a Version 7.0.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.0.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.0.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. You cannot restore a Version 7.0.1 installation to a server that has multiple IBM WebSphere MQ installations.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in group mqm.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
              ┌─-c─┐
►►──endmqm────┼─-w─┼──QmgrName──────────────────────────────────────────►◄
              ├─-i─┤
              └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 7.0.1 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Solaris: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “Solaris: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 390, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.0.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
4. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.

5. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.0.1.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                    ┌─-c─┐
      ►►──endmqm────┼─-w─┼──QmgrName─────────────────────────────────────────────────►◄
                    ├─-i─┤
                    └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.
   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Back up the queue manager.
5. Log in as root.

6. If you are running IBM WebSphere MQ Version 7.0.1.6 or later, optionally uninstall the current version of IBM WebSphere MQ. Note, that you carry out this step only if you are doing a single stage migration; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394
7. If you are running IBM WebSphere MQ Version 7.0.1.5 or earlier, uninstall the current version of IBM WebSphere MQ. If you require a side-by-side migration or multistage migration only, you must upgrade to IBM WebSphere MQ Version 7.0.1.6
8. Install IBM WebSphere MQ Version 7.5.

   See Installing WebSphere MQ Server on Solaris.
9. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                    ┌─-c─┐
      ►►──endmqm────┼─-w─┼──QmgrName──────────────────────────────────────────►◄
                    ├─-i─┤
                    └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.
   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Solaris: Restoring a Version 7.0.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.0.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.0.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. You cannot restore a Version 7.0.1 installation to a server that has multiple IBM WebSphere MQ installations.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.

   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.

   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.

   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 7.0.1 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 7.0.1 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, “Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5” on page 392, as a guide.
2. The migration paths to IBM WebSphere MQ Version 7.5 on Windows Vista are listed in Table 62 on page 510

Table 59. IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008: supported migration paths

From:	To:
IBM WebSphere MQ Version 6.0 plus fix pack 6.0.2.1 on Windows Vista or Windows Server 2008	IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008
IBM WebSphere MQ Version 7.0.1 on Windows XP Professional SP2	IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008

3. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
4. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 7.0.1. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5. For more information about backing up your system, see Backing up and restoring IBM WebSphere MQ queue manager data .
5. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
6. You must uninstall IBM WebSphere MQ Telemetry before installing IBM WebSphere MQ Version 7.5; see “Migrating IBM WebSphere MQ Telemetry from Version 7.0.1 to Version 7.5” on page 455.
7. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 7.0.1.

All the objects that you previously created are maintained. The components that were previously installed are preselected in the feature options when you install the new level. If you leave these components selected, you can keep them or reinstall them. If you clear any of these components, the installation process uninstalls them. By default, a typical migration installs only the same features that were installed in the previous version installation.

For example, if IBM WebSphere MQ Explorer was not installed in the Version 7.0.1 installation, it is not installed in a Version 7.5 installation. If you want IBM WebSphere MQ Explorer, select a custom installation, and select the IBM WebSphere MQ Explorer feature on the Features panel. If you do not want IBM WebSphere MQ Explorer, uninstall the IBM WebSphere MQ Explorer feature by selecting a custom installation. Then clear the IBM WebSphere MQ Explorer feature on the Features panel. For more information about how to uninstall features, see Modifying the installation using WebSphere MQ Installation Launchpad.

You can also migrate a Version 7.0.1 queue manager to IBM WebSphere MQ Version 7.5 on a system where a previous version has been uninstalled. The queue manager data must have been retained, or restored from a backup.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
                ┌─-c─┐
►►──endmqm──────┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
                ├─-i─┤
                └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Stop the MQSeries Service and exit the Service taskbar icon application.
6. Upgrade the IBM WebSphere MQ installation to Version 7.5 using one of these procedures:
   - Modifying the installation using WebSphere MQ Installation Launchpad
   - Silently modifying a WebSphere MQ server installation using **msiexec**
7. Reenter domain, user ID, and password information

   When the installation of IBM WebSphere MQ Version 7.5 completes, the Prepare WebSphere MQ Wizard starts automatically.

   On Windows Vista or Windows Server 2008 where UAC is enabled: If you rerun the Prepare WebSphere MQ Wizard, ensure that the wizard is run with Administrator privilege, otherwise the wizard might fail.
8. Move the queue manager to the new IBM WebSphere MQ installation. You need to carry out this step, only if you did not uninstall your current version of IBM WebSphere MQ.

   See "UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401 or "UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409 for further information.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Windows: Restoring a Version 7.0.1 queue manager from Version 7.5:

You can restore a queue manager to Version 7.0.1 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. For more information see Backing up and restoring IBM WebSphere MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 7.0.1 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. You cannot restore a Version 7.0.1 installation to a server that has multiple IBM WebSphere MQ installations.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
                ┌─-c─┐
►►──endmqm──────┼─-w─┼──QmgrName──────────────────────────────────────────►◄
                ├─-i─┤
                └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 7.0.1 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Migrating a IBM WebSphere MQ MQI client from Version 7.0.1 to Version 7.5

Migrate a IBM WebSphere MQ MQI client from Version 7.0.1 to Version 7.5 by doing the tasks in the following topics. Check IBM WebSphere MQ MQI client applications with Version 7.5 before migration. You must stop all IBM WebSphere MQ activity on the workstation, before upgrading the IBM WebSphere MQ MQI client. After upgrading the IBM WebSphere MQ MQI client, you must check the client channel configuration.

AIX: Migrating a IBM WebSphere MQ MQI client for AIX from Version 7.0.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "AIX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 384, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for AIX installation on a workstation; see Client installation procedure on an AIX workstation.
   - To upgrade a IBM WebSphere MQ MQI client for AIX installation on an AIX IBM WebSphere MQ server; see Installing a IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"AIX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 384
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Related information:

Client installation procedure on an AIX workstation
You can interactively install the IBM WebSphere MQ client for AIX using **smit**.

Installing WebSphere MQ MQI clients on the same machine as the server

HP-UX: Migrating a IBM WebSphere MQ MQI client for HP-UX from Version 7.0.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "HP-UX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 386, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.

3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for HP-UX installation on a workstation; see Client installation procedure on an HP-UX workstation.
   - To upgrade a IBM WebSphere MQ MQI client for HP-UX installation on an HP-UX IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"HP-UX: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 386
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Related information:

Client installation procedure on an HP-UX workstation

Linux: Migrating a IBM WebSphere MQ MQI client for Linux from Version 7.0.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Linux: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 388, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Linux installation on a workstation; see Client installation procedure on an Linux workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Linux installation on an Linux IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Linux: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 388
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Related information:

Client installation procedure on a Linux workstation
Installing a WebSphere MQ client on a 32 bit or 64 bit Linux system.

Solaris: Migrating a IBM WebSphere MQ MQI client for Solaris from Version 7.0.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Solaris: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 390, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Solaris installation on a workstation; see Client installation procedure on an Solaris workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Solaris installation on an Solaris IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Solaris: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 390
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Related information:

Client installation procedure on a Solaris workstation

Windows: Migrating a IBM WebSphere MQ MQI client for Windows from Version 7.0.1 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 392, as a guide.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Windows installation on a workstation; see Client installation procedure on an Windows workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Windows installation on an Windows IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Windows: Planning for migration from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 392
Create a migration plan to upgrade from Version 7.0.1 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 7.0.1 once you start it on Version 7.5.

Related information:

Client installation procedure on a Windcws workstation
This topic describes how to install IBM WebSphere MQ client on Windows systems. This procedure can be used for installing a first or a subsequent installation.

Restoring a Version 7.5 IBM WebSphere MQ MQI client and client connection to Version 7.0.1:

If you restore a Version 7.5 IBM WebSphere MQ MQI client to the Version 7.0.1 code level, you must undo the configuration changes manually.

About this task

It is unusual to restore earlier IBM WebSphere MQ MQI client libraries to a workstation. The principal tasks are listed in the following steps.

Procedure

1. End all IBM WebSphere MQ activity on the workstation.
2. Uninstall the Version 7.5 IBM WebSphere MQ MQI client code.
3. Follow the client installation procedure for the platform to install the Version 7.0.1IBM WebSphere MQ MQI client.
4. If you configured a Client Connection Definition Table (CCDT) from a Version 7.5 queue manager, revert to using a table created by a Version 7.0.1 queue manager.

   The CCDT must always be created by a queue manager on the same, or earlier, release to the client.

Migrating applications from Version 7.0.1 to Version 7.5

If you are installing Version 7.5 alongside Version 7.0.1, you must check what library is loaded by an application. If you must rebuild your application, build it with Version 7.5 libraries. Check the list of functions that have changed incompatibly from Version 7.0.1 to Version 7.5.

Procedure

1. Check the procedure to run your application loads the correct IBM WebSphere MQ Version 7.5 library.
   - On Windows, UNIX, and Linux platforms see "Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5."
   - For Java and JMS, check the correct `CLASSPATH` is set.

   For programs that are not written in Java, the library you require is different for 32 bit and 64 bit, and for client or server bindings.
2. Review the list of changes for changes that might affect your application.

   See "Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 561.
3. Do the following application migration tasks, if they are applicable to you.
   a. "Linux: Rebuilding a C++ application" on page 491

Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5:

No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Before you begin

To migrate applications from Version 7.0.1 to Version 7.5, you must know how the operating system loads a IBM WebSphere MQ library for an application. Is the load path fixed by the application, and can you set the path in an environment variable? It is not essential to know the name of the IBM WebSphere MQ library that the application loads. The library name does not change from Version 7.0.1 to Version 7.5, although the contents of the library do.

About this task

To migrate an application from Version 7.0.1 to Version 7.5, you do not have to recompile or relink the application, because the IBM WebSphere MQ libraries are compatible with later versions; see "Application compatibility and interoperability with later versions of WebSphere MQ" on page 356. You might have to configure the runtime environment differently, for the operating system to load the IBM WebSphere MQ Version 7.5 library. If you replaced Version 7.0.1 with Version 7.5, following the "Single-stage" approach; you do not need to do anything at all; see "UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394[9].

IBM WebSphere MQ Version 7.5 provides two commands to assist you in configuring the runtime environment, **`setmqinst`** and **`setmqenv`**. **`setmqinst`** sets the primary installation; see setmqinst. **`setmqenv`** initializes the command environment by setting environment variables; see setmqenv.

[9] If you changed the location of the IBM WebSphere MQ Version 7.0.1 libraries, or created symbolic links to the libraries, this statement might not hold true.

Procedure

How you configure the runtime environment depends on a number of factors, some of which apply to your configuration. Consider which of the following questions apply to your configuration.

1. Did you follow the build procedure documented in the IBM WebSphere MQ Version 7.0.1 product documentation?

   You might be following a different build procedure tailored to your development environment, or adapted from a development tool such as Microsoft Visual Studio.
2. How did you specify the load path Version 7.0.1?
3. Is the application is loaded by another environment, such as Eclipse, or an application server?

   You must modify the parameters that govern how the parent environment loads applications, not the way the parent environment is loaded.
4. Is the configuration for Windows, or UNIX and Linux?

   On Windows, the functions performed by an application might require that the queue manager it connects to is associated with the primary installation.
5. What constraints and requirements do you have on how the load path is specified in Version 7.5?

   Security rules might restrict the use of `LD_LIBRARY_PATH`.
6. Is Version 7.5 installed alongside Version 7.0.1?

   If Version 7.0.1 is installed:

   - You cannot make a Version 7.5 installation primary.
   - You cannot install Version 7.5 in the default installation path, that was referenced by applications in Version 7.0.1.

What to do next

Environment configuration on the Windows, and UNIX platforms is a little different. If you have followed the documented build procedure in Version 7.0.1, look at "Windows: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 485 or "UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488. These topics show the effects of using setmqinst and setmqenv commands to configure the operating system environment for the three migration scenarios listed in Related information.

Related tasks:

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

"Windows: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5"
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Related reference:

"Coexistence" on page 554
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Connecting applications in a multiple installation environment
On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

setmqenv

setmqinst

setmqm

Windows: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5:

Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Before you begin

Read "Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348 and "Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483 before starting this task.

Plan and install IBM WebSphere MQ for Windows Version 7.5, and remember the installation name and whether the installation was set to primary.

About this task

Windows searches numerous directories for load libraries, called `DLL`s; see Dynamic-Link Library Search Order.

The build procedure documented for IBM WebSphere MQ Version 7.0.1 applications is to place the IBM WebSphere MQ libraries to load before any other product libraries in the **`cl`** command. The IBM WebSphere MQ `.lib` libraries must be in the `PATH` environment variable you have specified at build time, and the `DLL` libraries at run time. The `PATH` variable is used by the application process to find the libraries it must load. If you have followed this build procedure, then the effect of installing Version 7.5 on the libraries that are loaded depends on the migration scenario; see Table 60.

Table 60. Windows configurations

Scenario Action	Version 7.5 replaces Version 7.0.1 in the same location “Single-stage”	Version 7.5 replaces Version 7.0.1 in a different location “Side-by-side”	Version 7.5 alongside Version 7.0.1 “Multi-stage”
`setmqinst`	**`setmqinst`** makes the Version 7.5 installation primary. The global `PATH` is changed to point to the Version 7.5 library and all Windows features work with Version 7.5$^{\text{See note}}$.		No Version 7.5 installation can be primary, because Version 7.0.1 is installed.
No other configuration actions	Library loading works correctly. The global `PATH` contains the location of the Version 7.5 libraries. Even if the Version 7.5 installation is not primary, library loading works correctly. The Version 7.5 libraries are in the same location as the Version 7.0.1 libraries were.	Library loading probably works correctly. The library loading might not work, if the application process locally modified the `PATH` to reference the location of the Version 7.0.1 libraries. A local setting of `PATH` might override the global `PATH` that is set by **`setmqinst`**.	The library loading continues to work with Version 7.0.1 correctly, nothing works with Version 7.5.
`setmqenv`	Library loading works correctly. **`setmqenv`** sets the local `PATH` correctly.		Library loading works correctly, both for Version 7.0.1 and Version 7.5. **`setmqenv`** sets the local `PATH` correctly for Version 7.5. But the Windows features that depend on the global path do not work correctly with Version 7.5 $^{\text{See note}}$. The correct Version 7.0.1 is loaded, because the Version 7.5 library loads the Version 7.0.1 library for queue managers that have not been migrated from Version 7.0.1.

Procedure

Identify which Version 7.5 installation the operating system is going to load IBM WebSphere MQ libraries from:

- If you have a multiple Version 7.5 installations to load from on a server, IBM WebSphere MQ checks that the installation the library was loaded from is the installation that is associated with any queue manager the application calls. IBM WebSphere MQ loads the correct library if the wrong library is loaded. It is necessary to configure only one runtime environment for all IBM WebSphere MQapplications.
- A typical choice is set the primary installation. Setting an installation to be primary places its library path in the global `PATH` variable.

- If you upgraded a Version 7.0.1 installation to Version 7.5, a link path to the Version 7.0.1 installation now points to a Version 7.5 installation. Applications that have a fixed linkage path to the Version 7.0.1 installation now load the Version 7.5 libraries. They are then switched to the installation that is associated with any queue manager they connect to.
- If you rebuild an application, it must link to a Version 7.5 installation.
- If an application uses COM or ActiveX it can connect to any queue manager as long as there is a primary installation and it is Version 7.5 or later.

  Note: If Version 7.0.1 is installed, COM or ActiveX server applications connect to queue managers associated only with the Version 7.0.1 installation. COM or ActiveX client applications are not affected by the limitation.
- If you are running the IBM WebSphere MQ .NET monitor in transactional mode, the queue manager it connects to must be the primary installation.

What to do next

If you add further Version 7.5 installations, you must decide which installation to make primary, if you have chosen to make any primary. As long as applications load IBM WebSphere MQ libraries from one of the Version 7.5 installations, such as the primary installation, they can connect to queue managers associated with any other Version 7.5 installation.

On Windows, you might build applications with different development tools. You must identify the property of the development tool that sets the `PATH` of the application that is being built, and not the properties of the tool itself. For example, if you are debugging with Microsoft Visual Studio, you can insert a call to **`setmqenv`** in the **`Environment`** property of the debugging section of the **`Configuration`** properties of a project.

A Windows application might call LoadLibrary and specify an explicit load path. You might build a side-by-side assembly and configure an explicit load path. If an application uses either of these mechanisms, and the Version 7.5 IBM WebSphere MQ library is not on the same path as the earlier release, you must recompile, or configure and relink your application to load the Version 7.5 libraries.

Related tasks:

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

Related reference:

"Coexistence" on page 554
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Connecting applications in a multiple installation environment
On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

setmqenv

setmqinst

setmqm

Features that can be used only with the primary installation on Windows
Some Windows operating-system features can be used only with the primary installation. This restriction is due to the central registration of interface libraries, which might conflict as a result of multiple versions of IBM WebSphere MQ being installed.

UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5:

Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Before you begin

Read "Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348 and "Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483 before starting this task.

Plan and install IBM WebSphere MQ for Windows Version 7.5, and remember the installation name and whether the installation was set to primary.

About this task

In Version 6.0 and Version 7.0.1, the documented build procedure for IBM WebSphere MQ applications is to include an explicit library path to the location of the IBM WebSphere MQ libraries, and to `/usr/lib`, in the link step of the compiler; see Figure 86 on page 489. The same build procedure is documented for Version 7.5. Previous versions of IBM WebSphere MQ, before Version 6.0, did not document a specific location to load run time libraries from.

```
gcc -m32 -o amqsput_32_r amqsput0.c -I/opt/mqm/inc -L/opt/mqm/lib
-Wl,-rpath=/opt/mqm/lib -Wl,-rpath=/usr/lib -lmqm_r -lpthread
```

Figure 86. Linux C server application, 32 bit, threaded compile and link Version 7.0.1

The documented build step for other UNIX platforms is similar. The examples in Table 61 are all based on Linux.

If you have followed this build procedure, then the effect of installing Version 7.5 on library loading depends on the migration scenario; see Table 61:

Table 61. UNIX and Linux configurations

Scenario	**Version 7.5 replaces Version 7.0.1 in the same location**	**Version 7.5 replaces Version 7.0.1 in a different location**	**Version 7.5 alongside Version 7.0.1**
Action	**"Single-stage"**	**"Side-by-side"**	**"Multi-stage"**
setmqinst	**setmqinst** makes the Version 7.5 installation primary. Symbolic links to the IBM WebSphere MQ link libraries are inserted into `/usr/lib`.		No Version 7.5 installation can be primary, because Version 7.0.1 is installed.
No other configuration actions	Library loading works correctly. Library loading works, even without the Version 7.5 installation being made primary, because the libraries are installed in `/opt/mqm/lib` and the application was built with the link option, `-rpath=/opt/mqm/lib`	Library loading works correctly. Library loading works, because the installation is primary, and the application was built with the link option, `-rpath=/usr/lib`.	The library loading continues to work with Version 7.0.1 correctly, nothing works with Version 7.5.
setmqenv, without setting the `-k` or `-l` options.	Library loading works correctly. **setmqenv** is unnecessary. Library loading works, because the libraries are installed in `/opt/mqm/lib` and the application was built with the link option, `-rpath=/opt/mqm/lib`.	Library loading works correctly. **setmqenv** is unnecessary. Library loading works, because the installation is primary, and the application was built with the link option, `-rpath=/usr/lib`.	The library loading continues to work with Version 7.0.1 correctly, nothing works with Version 7.5.
setmqenv, with the `-k` or `-l` options set.	Library loading works correctly.		Library loading works correctly, both for Version 7.0.1 and Version 7.5. The correct Version 7.0.1 is loaded, because the Version 7.5 library loads the Version 7.0.1 library for queue managers that have not been migrated from Version 7.0.1.
	The operating system finds the IBM WebSphere MQ library location set by **setmqenv**. **setmqenv** adds the location to `LD_LIBRARY_PATH`(`LIBPATH` on AIX. On HP-UX `LD_LIBRARY_PATH` is set, not `SHLIB_PATH`.). `LD_LIBRARY_PATH` is searched before paths set in the application or paths in the default search path. Not all applications can load a library using `LD_LIBRARY_PATH`. In which case the application works only if the library location is `/opt/mqm/lib` or `/usr/lib`		

Procedure

Identify which Version 7.5 installation the operating system is going to load IBM WebSphere MQ libraries from:

- If you have a multiple Version 7.5 installations to load from on a server, IBM WebSphere MQ checks that the installation the library was loaded from is the installation that is associated with any queue manager the application calls. IBM WebSphere MQ loads the correct library if the wrong library is loaded. It is necessary to configure only one runtime environment for all IBM WebSphere MQapplications.
- A typical choice is to set the primary installation. Setting an installation to be primary places symbolic links to the IBM WebSphere MQ libraries in `/usr/lib`. Applications built following the Version 6.0 or Version 7.0 instructions have an explicit link to `/usr/lib`. `/usr/lib` is also normally in the default library search path.
- If you upgraded a Version 7.0.1 installation to Version 7.5, a link path to the Version 7.0.1 installation now points to a Version 7.5 installation. Applications that have a fixed linkage path to the Version 7.0.1 installation now load the Version 7.5 libraries. They are then switched to the installation that is associated with any queue manager they connect to.
- If you rebuild an application, it must link to a Version 7.5 installation.
- If you set `LD_LIBRARY_PATH`, or `LIBPATH` on AIX, you must check that the application is able to use `LD_LIBRARY_PATH`. `setuid` or `setgid`, applications, or applications built in other ways, might ignore `LD_LIBRARY_PATH` for security reasons.

What to do next

If you add further Version 7.5 installations, you must decide which installation to make primary, if you have chosen to make any primary. As long as applications load IBM WebSphere MQ libraries from one of the Version 7.5 installations, such as the primary installation, they can connect to queue managers associated with any other Version 7.5 installation.

Related tasks:

"UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version" on page 394
Single-stage migration is the term used to describe replacing the only installation of IBM WebSphere MQ on a server, with a later release. Single stage migration is also known as **upgrading in place** or **in place upgrade**. Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM WebSphere MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

Related reference:

"Coexistence" on page 554
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information:

External library and control command links to primary installation on UNIX and Linux
On UNIX and Linux platforms the primary installation is the one to which links from the `/usr` file system are made. However, only a subset of those links created with previous releases are now made.

Connecting applications in a multiple installation environment
On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

setmqenv

setmqinst

setmqm

Loading WebSphere MQ Version 7.1 libraries
When deciding how to load IBM WebSphere MQ libraries, you need to consider a number of factors, including: your environment, whether you can change your existing applications, whether you want a primary installation, where IBM WebSphere MQ is installed, and whether the location of IBM WebSphere MQ is likely to change.

Linux: Rebuilding a C++ application:

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.

Before you begin

1. Check the required level of GCC for your distribution of Linux; see IBM WebSphere MQ requirements.
2. If you are using SSL or TLS, also check the required level of `libstdc++.so`.
3. Check whether the application requires rebuilding. Run the following command to display what version of `libstdc++.so` the application depends upon. If the result is less than `libstdc++.so.6`, you must rebuild your application.

   ```
   ldd ApplicationPath
   ```

About this task

The task describes the steps required to rebuild a Linux C++ IBM WebSphere MQ application. For more detailed instructions about building Linux applications for IBM WebSphere MQ; see Building your application on Linux

Procedure

1. Check that the required GCC library is installed correctly.

   Run one of the following commands:

   - Check the 32 bit library on an x86 Linux system:

     ```
     ls -l /usr/lib/libstdc++.so.6
     ```

   - Check the 64 bit library on any other Linux system.

     ```
     ls -l /usr/lib64/libstdc++.so.6
     ```

2. Check that the GCC compiler is at least at version 4.1.2

   Run the following command to display the version of GCC.

   ```
   gcc -v
   ```

3. Rebuild the application

   The commands to compile and link Linux C++ applications are described in Building 32-bit applications and Building 64-bit applications

What to do next

When you deploy your Linux C++ application, ensure that the same GCC runtime library is correctly installed on the run time system.

Migrating from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

The upgrade from Version 6.0 to Version 7.5 requires a full migration of queue managers. On platforms other than z/OS, if you start a queue manager on Version 7.5, you can not restore it to Version 6.0.

If you are using publish/subscribe, you must migrate from the Version 6.0 publish/subscribe broker to Version 7.5 publish/subscribe. You might also want to migrate a IBM WebSphere MQ MQI client to Version 7.5 to take advantage of new features. In some cases changes are necessary to for applications to run correctly in Version 7.5.

Migrating a queue manager from Version 6.0 to Version 7.5

The procedures for migrating a queue manager from Version 6.0 to Version 7.5 are listed by platform.

Before you begin

If you have installed early support program code on the server, you must delete all the queue managers created with the installation. Uninstall the code before proceeding with installing the production level code.

About this task

Migration is a complex task. It goes beyond upgrading the IBM WebSphere MQ code. The term "upgrading" applies to the process of installing a new code level. "Migrating" refers to the process of upgrading code and the to the task of migrating queue manager data, such as messages, queues, channels, and other resources. Migrating queue manager data is automatic.

Procedure

Queue manager migration follows this overall plan:

1. Review the WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
2. Back up your system. In particular back up the queue manager.

   You cannot run a queue manager on a previous version of IBM WebSphere MQ, once you start the queue manager on the new version.
3. Carry out the platform-specific migration or upgrade procedure.

   Consult "Planning IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5 migration by platform" on page 418.
   a. Upgrade IBM WebSphere MQ.
   b. Customize the configuration.
   c. Verify the installation.

      Create a queue manager to verify the installation. Verify applications and management tasks work with the new level of IBM WebSphere MQ before migrating existing queue managers. Consult "Characteristics of different types of upgrade" on page 343 to help you decide on the extent of testing required.
4. Perform any additional migration tasks that are required.

   If you are using publish/subscribe, you must migrate the publish/subscribe broker.

   If the queue manager is a member of a queue manager cluster, or queue-sharing group, migrate the other members of the cluster or group.

Related tasks:

"Planning IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5 migration by platform" on page 418
Before migrating from one version to another, read the planning topic for the platform you are interested in. Create your own migration plan based on the outline in the planning topic.

"Migration paths" on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

IBM WebSphere MQ requirements

AIX: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 6.0 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "AIX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 419, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 6.0. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5.
4. Remove the following file sets before migrating from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5; see "AIX: Removing message catalogs for IBM-850 code pages" on page 496.

mqm.msg.De_DE

mqm.msg.Es_ES

mqm.msg.Fr_FR

mqm.msg.It_IT

5. Remove SupportPac MA0Y, IBM WebSphere MQ bridge for HTTP; the function is incorporated into IBM WebSphere MQ Version 7.5.
6. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 6.0.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
          ┌─-c─┐
►►─endmqm─┼─-w─┼─QmgrName────────────────────────────────────────────►◄
          ├─-i─┤
          └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```

4. Back up the queue manager.
5. Log in as root, or switch to the superuser using the **su** command.
6. Uninstall the current version of IBM WebSphere MQ.
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on AIX.
8. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   Queue manager data is migrated by starting the queue manager.

   You must start IBM WebSphere MQ before you start any listeners.

What to do next

1. "AIX: Remedying the lack of shared storage to run 32 bit applications" on page 497.

Related tasks:

"AIX: Removing message catalogs for IBM-850 code pages" on page 496
Remove the IBM-850 code pages that are not supported by AIX at Version 5.3 or later.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

AIX: Remove Communications Server definitions of transaction programs:

An attempt to start a previously defined transaction program results in a SNA sense code of `084B6031`. In addition, error message AMQ9213 is returned. Remove the transaction program definitions.

Before you begin

You might have existing Communications Server definitions of transaction programs that can be started by IBM WebSphere MQ using the **snaadmin** command. **snaadmin** returns a sense code of `084B6031`. IBM WebSphere MQ returns an error message: AMQ9213: A communications error occurred.

About this task

Delete the transaction program definitions, and install and configure IBM Communications Server for AIX V6.3 or V6.4 to use SNA.

Procedure

1. Delete the definitions

   ```
   snaadmin delete_tp_load_info.tp_name=xxxxx
   snaadmin delete_tp.tp_name=xxxxx
   ```

2. Install IBM Communications Server for AIX V6.3 or V6.4

3. Recreate the transaction program using the following steps:
   a. Click **Services** > **APPC** > **Transaction programs**

      The TP Invocation window is opened.
   b. Type `MQSERIES` in the **Application TP** field.
   c. Clear the **Queue incoming Allocates** check box.
   d. Type `/usr/lpp/mqm/bin/amqcrs6a` in the **Full path to executable** field.
   e. Type `-n AIX -m` *`QmgrName`* in the **Arguments** field.
   f. Type `mqm` in the **User ID** and **Group ID** fields.
   g. Enter environment variables `APPCLLU=local LU` and `APPCTPN=Invokable TP` separated by the pipe "`|`" character in the **Environment** field.
   h. Click **OK**.

AIX: Removing message catalogs for IBM-850 code pages:

Remove the IBM-850 code pages that are not supported by AIX at Version 5.3 or later.

Before you begin

IBM WebSphere MQ for AIX requires a base AIX operating system at Version 5.3 or higher. From this level of operating system, the locales with IBM-850 code pages are no longer supported.

About this task

You must delete the following filesets, which contain message catalogs for IBM-850 code pages that are not included in IBM WebSphere MQ for AIX

```
mqm.msg.De_DE       (German)
mqm.msg.Es_ES       (Spanish)
mqm.msg.Fr_FR       (French)
mqm.msg.It_IT       (Italian)
```

Figure 87. Message catalogs that use the IBM-850 code page

Procedure

1. Use the **`lslpp`** command to list message catalogs.
2. Delete any of the file sets in Figure 87 that you find.
3. Use the **`locale`** command to find out if the language currently in use is from one of the withdrawn locales.
4. Change the `LANG` environment variable, or the `LC_MESSAGES` environment variable, or both, to see messages in your language.

Results

You cannot install IBM WebSphere MQ for AIX with a mixture of current, and earlier, message catalogs. The **`ifreqs`** command is set up in the packages to prevent installation with a mixture of message catalogs.

Related information:

Displaying messages in your national language
To display messages from a different national language message catalog, you must install the appropriate catalog and set the **LANG** environment variable.

AIX: Remedying the lack of shared storage to run 32 bit applications:

32 bit server applications types might fail to connect to the queue manager because of lack of shared storage. Solve this problem by increasing the amount of shared storage or running your application and the queue manager agent in separate processes. Shared storage problems do not affect IBM WebSphere MQ MQI client or 64 bit server applications.

Before you begin

This task applies only to 32 bit applications; the number of shared storage segments attached by 64 bit applications is practically unlimited.

About this task

On AIX:, 32 bit applications can attach only 10 shared storage segments concurrently. When an application connects to IBM WebSphere MQ, typically two storage segments are attached. However, the number of shared storage segments needed might increase depending on the workload of the queue manager.

Procedure

Choose one of the following two options to solve shared memory problems.

- Set the environment variable `EXTSHM=ON` in two places:
  1. Run `EXTSHM=ON` in the queue manager environment before running the **strmqm** command.
  2. Run `EXTSHM=ON` in the same environment as you are running your 32 bit applications.
- Specify `MQCNO_ISOLATED_BINDING` in the **MQCNO** parameter of the MQCONNX calls made by your applications.

  You must change a MQCONN call to a MQCONNX call. For JMS applications, you can set the **CONNOPT** property.

Related information:

Connecting to a queue manager using the MQCONNX call

Properties of WebSphere MQ classes for JMS objects

Shared memory on AIX
If certain application types fail to connect because of an AIX memory limitation, in most cases this can be resolved by setting the environment variable EXTSHM=ON.

AIX: Restoring a Version 6.0 queue manager from Version 7.5:

You can restore a queue manager to Version 6.0 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. See Backing up and restoring IBM WebSphere MQ queue manager data for more information.
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 6.0 in its current state.

The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.

3. If you are running on a server with multiple installations, you cannot restore a Version 6.0 queue manager to the same server.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

   ```
   endmqlsr -m QMgrName
   ```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.

   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.

   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.

   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 6.0 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

HP-UX: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 6.0 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "HP-UX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 421, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 6.0. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5.
4. Remove SupportPac MA0Y, IBM WebSphere MQ bridge for HTTP; the function is incorporated into IBM WebSphere MQ Version 7.5.
5. Remove SupportPac MA6P, IBM WebSphere MQ XA libraries for use with TX-series on HP-UX: for IPF; the function is incorporated into IBM WebSphere MQ Version 7.5.
6. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 6.0.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.

3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.

   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root, or switch to the superuser using the **su** command.
6. Uninstall WebSphere MQ version 6.0.
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on HP-UX.
8. Start the queue manager.

```
strmqm QmgrName
```

Queue manager data is migrated by starting the queue manager.

You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

HP-UX: Restoring a Version 6.0 queue manager from Version 7.5:

You can restore a queue manager to Version 6.0 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. See Backing up and restoring IBM WebSphere MQ queue manager data for more information.
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 6.0 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple installations, you cannot restore a Version 6.0 queue manager to the same server.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
              ┌─-c─┐
▶▶──endmqm────┼─-w─┼──QmgrName────────────────────────────────────────────────▶◀
              ├─-i─┤
              └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 6.0 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Linux: Cleaning up after using the rpm freshen or upgrade options:

The use of **rpm** upgrade or freshen options is not supported. If you use the options, follow this cleanup procedure, and then install following the correct steps.

Before you begin

You have attempted to upgrade IBM WebSphere MQ for Linux using `rpm -U` or `rpm -F`

About this task

By using the freshen or upgrade options, you might have deleted your old IBM WebSphere MQ package entries from the **rpm** database without removing the product from your system. You might also have partially installed IBM WebSphere MQ

Procedure

Follow these steps to clean up your system.

1. Find out which IBM WebSphere MQ MQ packages still have entries in your RPM database.
   ```
   rpm -qa | grep MQSeries
   ```
2. Remove all remaining IBM WebSphere MQ packages from your system.
   ```
   rpm -e package-name
   ```
3. Remove the `/opt/mqm` directory.
   ```
   rm -rf /opt/mqm
   ```

Linux: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 6.0 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "Linux: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 423, as a guide.
2. If Linux is running on System X, and your queue manager is 32 bit, and you want to migrate it to 64 bit, you must migrate to IBM WebSphere MQ Version 7.0.1 first. 32 to 64 bit migration is not supported by IBM WebSphere MQ Version 7.5
3. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
4. If you are considering switching IBM WebSphere MQ for Linux System x from a 32 bit to a 64 bit installation do not migrate to Version 7.5. Your last chance to switch to 64 bit is the migration to Version 7.0.1.
5. Do not use the **rpm** `-U` (upgrade) or `-F` (freshen) options to upgrade IBM WebSphere MQ for Linux; see "Linux: Cleaning up after using the **rpm** freshen or upgrade options" on page 439.
6. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 6.0. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5.
7. Remove SupportPac MA0Y, IBM WebSphere MQ bridge for HTTP; the function is incorporated into IBM WebSphere MQ Version 7.5.
8. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 6.0.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
               ┌─-c─┐
►►──endmqm─────┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
               ├─-i─┤
               └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root, or switch to the superuser using the **su** command.
6. Uninstall WebSphere MQ version 6.0.
7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on Linux.
8. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   Queue manager data is migrated by starting the queue manager.

   You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Linux: Restoring a Version 6.0 queue manager from Version 7.5:

You can restore a queue manager to Version 6.0 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. See Backing up and restoring IBM WebSphere MQ queue manager data for more information.
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 6.0 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple installations, you cannot restore a Version 6.0 queue manager to the same server.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **`MQSC`** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
           ┌─-c─┐
►►─endmqm──┼─-w─┼─QmgrName──────────────────────────────────────────────►◄
           ├─-i─┤
           └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.
   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.
   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.
   d. Restore the queue manager data from the backup taken before installing Version 7.5.
5. Restart the Version 6.0 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Solaris: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 6.0 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "Solaris: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 425, as a guide.
2. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
3. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 6.0. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5.
4. Remove SupportPac MA0Y, IBM WebSphere MQ bridge for HTTP; the function is incorporated into IBM WebSphere MQ Version 7.5.

5. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 6.0.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **`MQSC`** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **`endmqm`** command to stop each running queue manager associated with this installation.

      ```
                    ┌─-c─┐
      ►►──endmqm────┼─-w─┼──QmgrName──────────────────────────────────────────►◄
                    ├─-i─┤
                    └─-p─┘
      ```

      The **`endmqm`** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **`endmqm`** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Back up the queue manager.
5. Log in as root, or switch to the superuser using the **`su`** command.
6. Uninstall WebSphere MQ version 6.0.

7. Install IBM WebSphere MQ Version 7.5

   See Installing WebSphere MQ Server on Solaris.

8. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   Queue manager data is migrated by starting the queue manager.

   You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Solaris: Restoring a Version 6.0 queue manager from Version 7.5:

You can restore a queue manager to Version 6.0 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. See Backing up and restoring IBM WebSphere MQ queue manager data for more information.
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 6.0 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple installations, you cannot restore a Version 6.0 queue manager to the same server.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.

3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

    dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

    dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                      ┌-c┐
      ►►──endmqm──────┼-w┼──QmgrName──────────────────────────────────────►◄
                      ├-i┤
                      └-p┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.

   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.

   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.

   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 6.0 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Windows: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5:

Follow these instructions to migrate a queue manager from Version 6.0 to Version 7.5.

Before you begin

1. Create a migration plan. Use the planning task, "Windows: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 427, as a guide.

2. The migration paths to IBM WebSphere MQ Version 7.5 on Windows Vista are listed in Table 62

Table 62. IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008: supported migration paths

From:	To:
IBM WebSphere MQ Version 6.0 plus fix pack 6.0.2.1 on Windows Vista or Windows Server 2008	IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008
IBM WebSphere MQ Version 7.0.1 on Windows XP Professional SP2	IBM WebSphere MQ Version 7.5 on Windows Vista or Windows Server 2008

3. Review the IBM WebSphere MQ system requirements for Version 7.5; see IBM WebSphere MQ requirements.
4. Back up your system before you install IBM WebSphere MQ Version 7.5 over Version 6.0. Once you have started a queue manager you cannot revert to the previous release. If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by IBM WebSphere MQ Version 7.5.
5. Remove SupportPac MA0Y, IBM WebSphere MQ bridge for HTTP; the function is incorporated into IBM WebSphere MQ Version 7.5.
6. Review any other installed SupportPacs for their applicability to IBM WebSphere MQ Version 7.5.
7. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to IBM WebSphere MQ Version 7.5 from Version 6.0.

All the objects that you previously created are maintained. The components that were previously installed are preselected in the feature options when you install the new level. If you leave these components selected, you can keep them or reinstall them. If you clear any of these components, the installation process uninstalls them. By default, a typical migration installs only the same features that were installed in the previous version installation.

For example, if IBM WebSphere MQ Explorer was not installed in the Version 6.0 installation, it is not installed in a Version 7.5 installation. If you want IBM WebSphere MQ Explorer, select a custom installation, and select the IBM WebSphere MQ Explorer feature on the Features panel. If you do not want IBM WebSphere MQ Explorer, uninstall the IBM WebSphere MQ Explorer feature by selecting a custom installation. Then clear the IBM WebSphere MQ Explorer feature on the Features panel. For more information about how to uninstall features, see Modifying the installation using WebSphere MQ Installation Launchpad.

You can also migrate a Version 6.0 queue manager to IBM WebSphere MQ Version 7.5 on a system where a previous version has been uninstalled. The queue manager data must have been retained, or restored from a backup.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.

a. Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

   dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

   dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

   ```
                ┌─-c─┐
   ►►──endmqm───┼─-w─┼──QmgrName──────────────────────────────────────────────────►◄
                ├─-i─┤
                └─-p─┘
   ```

   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

   You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

   Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

   ```
   endmqlsr -m QMgrName
   ```

4. Back up the queue manager.
5. Stop the MQSeries Service and exit the Service taskbar icon application.
6. Upgrade the IBM WebSphere MQ installation to Version 7.5 using one of these procedures:
   - Modifying the installation using WebSphere MQ Installation Launchpad
   - Silently modifying a WebSphere MQ server installation using **msiexec**
7. Reenter domain, user ID, and password information

   When the installation of IBM WebSphere MQ Version 7.5 completes, the Prepare WebSphere MQ Wizard starts automatically.

   On Windows Vista or Windows Server 2008 where UAC is enabled: If you rerun the Prepare WebSphere MQ Wizard, ensure that the wizard is run with Administrator privilege, otherwise the wizard might fail.
8. Start the queue manager.

   ```
   strmqm QmgrName
   ```

   Queue manager data is migrated by starting the queue manager.

   You must start IBM WebSphere MQ before you start any listeners.

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

Windows: Restoring a Version 6.0 queue manager from Version 7.5:

You can restore a queue manager to Version 6.0 from Version 7.5, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to Version 7.5. See Backing up and restoring IBM WebSphere MQ queue manager data for more information.
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to Version 6.0 in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple installations, you cannot restore a Version 6.0 queue manager to the same server.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   1. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   2. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
          ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────────────►◄
          ├─-i─┤
          └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM WebSphere MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM WebSphere MQ.

   a. Uninstall the Version 7.5 installation.

   b. Reinstall IBM WebSphere MQ from a manufacturing refresh.

   c. Apply the fix pack and interim fixes that restore IBM WebSphere MQ to its previous level.

   d. Restore the queue manager data from the backup taken before installing Version 7.5.

5. Restart the Version 6.0 queue manager.

Related information:

Backing up and restoring a queue manager
Backing up queue managers and queue manager data.

Migrating a IBM WebSphere MQ MQI client from Version 6.0 to Version 7.5

Migrate a IBM WebSphere MQ MQI client from Version 6.0 to Version 7.5 by doing the tasks in the following topics. Check IBM WebSphere MQ MQI client applications with Version 7.5 before migration. You must stop all IBM WebSphere MQ activity on the workstation, before upgrading the IBM WebSphere MQ MQI client. After upgrading theIBM WebSphere MQ MQI client, you must check the client channel configuration.

AIX: Migrating a IBM WebSphere MQ MQI client for AIX from Version 6.0 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "AIX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 419, as a guide.
2. Find out whether any client connections are using channel exits.
3. Find out whether the request-response time is critical for any client, to within about a 15% degradation.
4. Find out whether any client applications call MQPUT1 within sync point. Modify the reason code checking of any client applications that call MQPUT1 within sync point.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for AIX installation on a workstation; see Client installation procedure on an AIX workstation.
   - To upgrade a IBM WebSphere MQ MQI client for AIX installation on an AIX IBM WebSphere MQ server; see Installing a IBM WebSphere MQ MQI client on the same computer as the server.
5. Review the setting of the new channel attribute, **SHARECNV**.

   The default setting of **SHARECNV** for Version 7.5 server-connection channel is 10. 10 client instances are allocated per channel. On average, sharing a channel conversation increases the request-response time for an individual instance by 15%.

   Set **SHARECNV** to `0` or 1 to stop sharing the channel instance.

   The default behavior of a client or server connection channel exit has changed in Version 7.5. Set **SHARECNV** to `0` to restore Version 6.0 behavior.

   You can control the behavior of individual applications by setting MQCONNX option, `MQCNO_NO_CONV_SHARING` or `MQCNO_ALL_CONVS_SHARE`.
6. Check the client configuration settings.

   The client upgrade in step 4 migrated client configuration settings from `qm.ini` and `mqs.ini` into `mqclient.ini`. Check the settings.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"AIX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 419
Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Related information:

Client installation procedure on an AIX workstation
You can interactively install the IBM WebSphere MQ client for AIX using **smit**.

Installing WebSphere MQ MQI clients on the same machine as the server

"MQI client: Client configuration stanzas moved into a new configuration file" on page 600
Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

HP-UX: Migrating a IBM WebSphere MQ MQI client for HP-UX from Version 6.0 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "HP-UX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 421, as a guide.
2. Find out whether any client connections are using channel exits.
3. Find out whether the request-response time is critical for any client, to within about a 15% degradation.
4. Find out whether any client applications call MQPUT1 within sync point. Modify the reason code checking of any client applications that call MQPUT1 within sync point.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for HP-UX installation on a workstation; see Client installation procedure on an HP-UX workstation.
   - To upgrade a IBM WebSphere MQ MQI client for HP-UX installation on an HP-UX IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.
5. Review the setting of the new channel attribute, **SHARECNV**.

The default setting of **SHARECNV** for Version 7.5 server-connection channel is 10. 10 client instances are allocated per channel. On average, sharing a channel conversation increases the request-response time for an individual instance by 15%.

Set **SHARECNV** to `0` or `1` to stop sharing the channel instance.

The default behavior of a client or server connection channel exit has changed in Version 7.5. Set **SHARECNV** to `0` to restore Version 6.0 behavior.

You can control the behavior of individual applications by setting MQCONNX option, `MQCNO_NO_CONV_SHARING` or `MQCNO_ALL_CONVS_SHARE`.

6. Check the client configuration settings.

   The client upgrade in step 4 on page 514 migrated client configuration settings from `qm.ini` and `mqs.ini` into `mqclient.ini`. Check the settings.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"HP-UX: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 421
Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Related information:

Client installation procedure on an HP-UX workstation

Installing WebSphere MQ MQI clients on the same machine as the server

"MQI client: Client configuration stanzas moved into a new configuration file" on page 600
Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

Linux: Migrating a IBM WebSphere MQ MQI client for Linux from Version 6.0 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Linux: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 423, as a guide.
2. Find out whether any client connections are using channel exits.
3. Find out whether the request-response time is critical for any client, to within about a 15% degradation.
4. Find out whether any client applications call MQPUT1 within sync point. Modify the reason code checking of any client applications that call MQPUT1 within sync point.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Linux installation on a workstation; see Client installation procedure on a Linux workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Linux installation on an Linux IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.
5. Review the setting of the new channel attribute, **SHARECNV**.

   The default setting of **SHARECNV** for Version 7.5 server-connection channel is 10. 10 client instances are allocated per channel. On average, sharing a channel conversation increases the request-response time for an individual instance by 15%.

   Set **SHARECNV** to `0` or `1` to stop sharing the channel instance.

   The default behavior of a client or server connection channel exit has changed in Version 7.5. Set **SHARECNV** to `0` to restore Version 6.0 behavior.

   You can control the behavior of individual applications by setting MQCONNX option, `MQCNO_NO_CONV_SHARING` or `MQCNO_ALL_CONVS_SHARE`.
6. Check the client configuration settings.

   The client upgrade in step 4 on page 514 migrated client configuration settings from `qm.ini` and `mqs.ini` into `mqclient.ini`. Check the settings.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Linux: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 423
Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Related information:

Client installation procedure on a Linux workstation

Installing WebSphere MQ MQI clients on the same machine as the server

"MQI client: Client configuration stanzas moved into a new configuration file" on page 600
Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

Solaris: Migrating a IBM WebSphere MQ MQI client for Solaris from Version 6.0 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Solaris: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 425, as a guide.
2. Find out whether any client connections are using channel exits.
3. Find out whether the request-response time is critical for any client, to within about a 15% degradation.
4. Find out whether any client applications call MQPUT1 within sync point. Modify the reason code checking of any client applications that call MQPUT1 within sync point.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Solaris installation on a workstation; see Client installation procedure on a Solaris workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Solaris installation on an Solaris IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.
5. Review the setting of the new channel attribute, **SHARECNV**.

   The default setting of **SHARECNV** for Version 7.5 server-connection channel is 10. 10 client instances are allocated per channel. On average, sharing a channel conversation increases the request-response time for an individual instance by 15%.

   Set **SHARECNV** to `0` or 1 to stop sharing the channel instance.

   The default behavior of a client or server connection channel exit has changed in Version 7.5. Set **SHARECNV** to `0` to restore Version 6.0 behavior.

   You can control the behavior of individual applications by setting MQCONNX option, `MQCNO_NO_CONV_SHARING` or `MQCNO_ALL_CONVS_SHARE`.
6. Check the client configuration settings.

   The client upgrade in step 4 on page 514 migrated client configuration settings from `qm.ini` and `mqs.ini` into `mqclient.ini`. Check the settings.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Solaris: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 425
Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Related information:

Client installation procedure on a Solaris workstation

Installing WebSphere MQ MQI clients on the same machine as the server

"MQI client: Client configuration stanzas moved into a new configuration file" on page 600
Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

Windows: Migrating a IBM WebSphere MQ MQI client for Windows from Version 6.0 to Version 7.5:

Before migrating a IBM WebSphere MQ MQI client, create a migration plan. Stop all IBM WebSphere MQ activity on the client workstation. Upgrade the IBM WebSphere MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "Windows: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 427, as a guide.
2. Find out whether any client connections are using channel exits.
3. Find out whether the request-response time is critical for any client, to within about a 15% degradation.
4. Find out whether any client applications call MQPUT1 within sync point. Modify the reason code checking of any client applications that call MQPUT1 within sync point.

Procedure

1. Review the WebSphere MQ system requirements for Version 7.5.

   See IBM WebSphere MQ requirements.
2. Review all the changes in WebSphere MQ that affect you.

   See changes from "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.
3. End all IBM WebSphere MQ activity on the workstation.
4. Upgrade the client.
   - To upgrade a IBM WebSphere MQ MQI client for Windows installation on a workstation; see Client installation procedure on a Windows workstation.
   - To upgrade a IBM WebSphere MQ MQI client for Windows installation on an Windows IBM WebSphere MQ server; see Installing an IBM WebSphere MQ MQI client on the same computer as the server.
5. Review the setting of the new channel attribute, **`SHARECNV`**.

The default setting of **SHARECNV** for Version 7.5 server-connection channel is 10. 10 client instances are allocated per channel. On average, sharing a channel conversation increases the request-response time for an individual instance by 15%.

Set **SHARECNV** to 0 or 1 to stop sharing the channel instance.

The default behavior of a client or server connection channel exit has changed in Version 7.5. Set **SHARECNV** to 0 to restore Version 6.0 behavior.

You can control the behavior of individual applications by setting MQCONNX option, MQCNO_NO_CONV_SHARING or MQCNO_ALL_CONVS_SHARE.

6. Check the client configuration settings.

   The client upgrade in step 4 on page 514 migrated client configuration settings from qm.ini and mqs.ini into mqclient.ini. Check the settings.

What to do next

Complete the tasks in your migration plan, such as verifying IBM WebSphere MQ MQI client applications work correctly with Version 7.5.

Related tasks:

"Windows: Planning for migration from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 427
Create a migration plan to upgrade from Version 6.0 to Version 7.5. You must back up your queue manager data before you migrate to Version 7.5. You cannot revert a queue manager to Version 6.0 once you start it on Version 7.5.

Related information:

Client installation procedure on a Windows workstation

amqmdain: Display and modify WebSphere MQ Windows registry entries

Installing WebSphere MQ MQI clients on the same machine as the server

"MQI client: Client configuration stanzas moved into a new configuration file" on page 600
Client configuration information is moved from existing configuration stanzas into a new configuration file, mqclient.ini.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

Restoring a Version 7.5 IBM WebSphere MQ MQI client and client connection to Version 6.0:

If you restore a Version 7.5 IBM WebSphere MQ MQI client to the Version 6.0 code level, you must undo the configuration changes manually.

About this task

It is unusual to restore earlier IBM WebSphere MQ MQI client libraries to a workstation. The principal tasks are listed in the following steps.

Procedure

1. End all IBM WebSphere MQ activity on the workstation.
2. Uninstall the Version 7.5 IBM WebSphere MQ MQI client code.

3. Follow the client installation procedure for the platform to install the Version 6.0IBM WebSphere MQ MQI client.
4. Restore configuration settings from `mqclient.ini` to `qm.ini`, `mqs.ini`, or to the Windows registry.
5. If you configured a Client Channel Definition Table (CCDT) from a Version 7.5 queue manager, revert to using a table created by a Version 6.0 queue manager.

   The CCDT must always be created by a queue manager on the same, or earlier, release to the client.

Related information:

amqmdain: Display and modify WebSphere MQ Windows registry entries

Installing WebSphere MQ MQI clients on the same machine as the server

"MQI client: Client configuration stanzas moved into a new configuration file" on page 600
Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

"MQI client: Default behavior of client-connection and server-connection" on page 601
The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

Migrating publish/subscribe from Version 6.0 to Version 7.5

Migrate a Version 6.0 publish/subscribe broker to a Version 7.5 queue manager, migrating publications, subscriptions, and the publish/subscribe configuration.

Before you begin

1. You have migrated the queue manager from Version 6.0 to Version 7.5.
2. The IBM WebSphere MQ publish/subscribe broker in Version 6.0 was connected to this queue manager.

About this task

You can migrate the Version 6.0 publish/subscribe configuration to Version 7.5 at any time after you have started the queue manager. Initially, after you have migrated the queue manager, the queue manager **PSMODE** attribute is set to `COMPAT`. You can migrate the publish/subscribe broker only if **PSMODE** is set to `COMPAT`. If you are running the queue manager with **PSMODE** set to `ENABLED` or `DISABLED`, change the setting to `COMPAT`.

The `COMPAT` setting means integrated publish/subscribe is started, but queued publish/subscribe is not. The setting defaults this way in case a WebSphere Message Broker or WebSphere Event Broker version 6.0 or 6.1 is connected to the queue manager. It might be running its publish/subscribe broker. Migrating the queue manager to Version 7.5 then does not interfere with running the WebSphere Message Broker or WebSphere Event Broker publish/subscribe broker. Typically, neither WebSphere Message Broker or WebSphere Event Broker are a concern to you.

Procedure

1. Start the queue manager.

   ```
   strmqm QmgrName
   ```
2. Check **PSMODE** is set to `COMPAT`.
   -

```
echo DISPLAY QMGR PSMODE | runmqsc QmgrName
```

Figure 88. Windows

•

```
echo "DISPLAY QMGR PSMODE" | runmqsc QmgrName
```

Figure 89. UNIX or Linux

3. Migrate the publish/subscribe broker from Version 6.0 to Version 7.5.
   `strmqbrk -m QmgrName`
4. Inspect the migration log for any warnings.

   The migration log file is in the queue manager data directory.

   If migration would create a duplicate object, a warning is written to the log.
5. Check that **`PSMODE`** is now set to `ENABLED`.

   The migration process automatically sets **`PSMODE`** to `ENABLED` to allow existing queued publish/subscribe applications to function with the migrated queue manager.

What to do next

1. Alter publish/subscribe monitoring; meta topics are no longer supported.
2. Monitor applications for changes in behavior that result from differences in the publish/subscribe models.
3. Migrate other queue managers in a hierarchy.
4. Convert a publish/subscribe hierarchy into a cluster.
5. Migrate WebSphere Message Broker to version 7.0.

Related concepts:

"Publish/Subscribe migration from Version 6.0" on page 367

"Migration of the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker" on page 369
The publish/subscribe broker used in WebSphere Message Broker version 6.0 and 6.1 is replaced by the integrated publish/subscribe broker in IBM WebSphere MQ. IBM WebSphere MQ replaces WebSphere Event Broker version 6.0. The migration to IBM WebSphere MQ has two parts. The migration of publish/subscribe configuration data, and conversion of publish/subscribe collectives to publish/subscribe clusters.

Related reference:

"Publish/Subscribe: Application migration" on page 634
The Version 6.0 IBM WebSphere MQ publish/subscribe command message interface has been superseded by the IBM WebSphere MQ publish/subscribe programming interface. It is your choice whether to migrate existing applications to the new interface. The following topics provide reference information to help you migrate applications to the publish/subscribe programming interface.

"Publish/subscribe changes" on page 633
Publish/subscribe function is now performed by the queue manager, rather than by a separate publish/subscribe broker. When you upgrade a IBM WebSphere MQ Version 6.0 queue manager, publish/subscribe function is not automatically migrated. You must migrate publish/subscribe using the **`strmqbrk`** command. After migration, with the differences noted in the following topics, existing Version 6.0 publish/subscribe programs work without change.

Migrating a IBM WebSphere MQ Version 6.0 publish/subscribe hierarchy to a IBM WebSphere MQ Version 7.5 hierarchy:

Migrate queue managers in a Version 6.0 publish/subscribe hierarchy to Version 7.5, a queue manager, one server at a time.

Before you begin

You can migrate a Version 6.0 publish/subscribe hierarchy to Version 7.5 one server at a time. You do not need to migrate the whole hierarchy at once. The hierarchy continues to work with some queue managers at Version 6.0, and some at Version 7.5.

Consider how you are going to upgrade multiple queue managers on one server to Version 7.5. Are you going to upgrade the queue managers one at a time, or can you follow the simpler course and upgrade them altogether? All the queue managers on a server share the same IBM WebSphere MQ libraries and therefore are at the same release level. You cannot upgrade some queue managers on the same server to Version 7.5, leaving others on Version 6.0.

Consider these two alternatives, if you are running multiple queue managers on the same server.

1. The simpler approach to migrating publish/subscribe to Version 7.5 is to upgrade IBM WebSphere MQ on the server to Version 7.5.

   Migrate the publish/subscribe broker for each queue manager on the server to Version 7.5 publish/subscribe, one queue manager at a time.
2. If you want to upgrade the queue managers to Version 7.5 one at a time, you must perform a "side-by-side" upgrade rather than an "in-place" upgrade.

   In a side-by-side upgrade:

   a. Install IBM WebSphere MQ Version 7.5 on a new server.

   b. Transfer a queue manager to the new server.

   c. Migrate its Version 6.0 broker to Version 7.5.

   d. Repeat, one queue manager at a time, until all the queue managers are running on the new server, and none are left on the old server.

About this task

In this task, we assume that you have already upgraded a queue manager to Version 7.5. You now want to migrate the publish/subscribe broker to Version 7.5.

You might be migrating the first publish/subscribe broker in the hierarchy, or the last. Follow the same procedure. At any point in the conversion, of all the publish/subscribe brokers in the hierarchy, the hierarchy, and your applications continue to work.

Procedure

1. Start the queue manager.

   ```
   strmqm QMgrName
   ```
2. Run the **strmqbrk** command to migrate all publish/subscribe configuration data for the queue manager to IBM WebSphere MQ Version 7.5.

   ```
   strmqbrk -m QMgrName
   ```
3. Check the migration log to verify that the migration was successful.

   The migration log is in the queue manager data directory: the default locations are `/var/mqm/qmgrs/`*QMgrName*`/psmigr.log` on UNIX or `C:\Program Files\IBM\WebSphereMQ\qmgrs\`*QMgrName*`\psmigr.log` on Windows.

4. **strmqbrk** starts queued publish/subscribe when it completes, by switching the queue manager attribute **PSMODE** from its default value of COMPAT to ENABLED.

Migrating applications from Version 6.0 to Version 7.5

Check the list of functions that have changed incompatibly from Version 6.0 to Version 7.5. If you must rebuild your application, build it with either Version 6.0 or Version 7.5 libraries.

Procedure

1. Check the procedure to run your application loads the correct IBM WebSphere MQ Version 7.5 library.
   - Windows UNIX Linux On Windows, UNIX and Linux platforms check the path, to pick up the correct library provided by IBM WebSphere MQ.
   - For Java and JMS, check the correct CLASSPATH is set.

     For programs that are not written in Java, the library you require is different for 32 bit and 64 bit, and for client or server bindings.
2. Review the list of changes for changes that might affect your application.

   See "Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 596.

   Note the following changes:
   a. "Apache Axis shipped with IBM WebSphere MQ updated from version 1.1 to 1.4" on page 596
   b. "MQI client: MQPUT1 sync point behavior change" on page 602
   c. "MQI reason code changes" on page 626
   d. "MQRFH2 migration, message properties, and property folders" on page 627
   e. "Publish/subscribe changes" on page 633
   f. SSL or TLS Authority Information Access (AIA) checked by default
   g. "Preferred alternatives: Consider replacing these Version 6.0 functions with their Version 7.5 alternatives" on page 659
   h. "Migrating JMS applications from Version 6.0 to Version 7.5"

   If you must rebuild your application, you can rebuild it with either Version 6.0 or Version 7.5 libraries.

   If you rebuild it with Version 6.0 libraries, you can run it on either Version 6.0 or Version 7.5. It cannot access new Version 7.5 functions.

   If you rebuild it with Version 7.5 libraries, you must not load Version 6.0 libraries. If it is a server application it must connect only to a Version 7.5 or later queue manager. If it is a client application, it can connect to any level of queue manager.

Migrating JMS applications from Version 6.0 to Version 7.5:

Existing IBM WebSphere MQ classes for JMS applications continue to run without any manual intervention after you upgrade your queue manager to IBM WebSphere MQ Version 7.0. In some circumstances, you must modify the application, or the application properties, for an application to continue to run correctly.

Before you begin

1. The queue manager must be running Version 7.5.
2. IBM WebSphere MQ classes for JMS contains both server and client bindings. The version of IBM WebSphere MQ classes for JMS by the Java application must be Version 7.5.
3. If your application is loaded by WebSphere Application Server, check the correct IBM WebSphere MQ classes for JMS are loaded; see "JMS: Integration of IBM WebSphere MQ classes for JMS with WebSphere Application Server" on page 618.

About this task

In many cases, JMS applications that connected to IBM WebSphere MQ Version 6.0, run correctly without any manual changes when they are connected to IBM WebSphere MQ Version 7.5. Details about the changes that might affect the correct behavior of a JMS application that has been migrated from Version 6.0 to Version 7.5 are listed in "JMS and Java changes" on page 613. This task steps you through the main changes that might affect your application.

Procedure

Review and modify your application if it has one or more of these characteristics.

1. It runs under the control of WebSphere Application Server

   See "JMS: Integration of IBM WebSphere MQ classes for JMS with WebSphere Application Server" on page 618.
2. It parses or tests exception messages returned by Throwable.getMessage or JMSException.getErrorCode.

   See "JMS: Error code and exception message changes" on page 616 and "Java MQMessage.resizeBuffer exception change" on page 613.
3. It publishes or subscribes.

   See "JMS: Messaging provider" on page 622 and "JMS: Migration of subscriptions to named streams" on page 622.
4. It interacts with other applications that parse the `MQRFH2` in the message.

   See "JMS: Receiving `MQRFH2` headers in messages sent from JMS applications" on page 624 and "MQRFH2 migration, message properties, and property folders" on page 627.
5. It is a channel exit.

   See "JMS: Channel exits written in C and C++ called from JMS have changed behavior" on page 613.
6. It uses tracing.

   See "JMS: Tracing and error reporting" on page 625.
7. It uses property names containing dots.

   See "JMS: Dots in property names" on page 616.

Related reference:

"JMS and Java changes" on page 613
WebSphere MQ classes for JMS has been rewritten to integrate it with IBM WebSphere MQ queue manager internal interfaces. To a large extent, its functional behavior remains the same. Some changes affect the way existing JMS and Java applications behave. The differences are noted in the following topics.

Linux: Rebuilding a C++ application:

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.

Before you begin

1. Check the required level of GCC for your distribution of Linux; see IBM WebSphere MQ requirements.
2. If you are using SSL or TLS, also check the required level of `libstdc++.so`.
3. Check whether the application requires rebuilding. Run the following command to display what version of `libstdc++.so` the application depends upon. If the result is less than `libstdc++.so.6`, you must rebuild your application.

   ```
   ldd ApplicationPath
   ```

About this task

The task describes the steps required to rebuild a Linux C++ IBM WebSphere MQ application. For more detailed instructions about building Linux applications for IBM WebSphere MQ; see Building your application on Linux

Procedure

1. Check that the required GCC library is installed correctly.

   Run one of the following commands:

   - Check the 32 bit library on an x86 Linux system:

     ```
     ls -l /usr/lib/libstdc++.so.6
     ```

   - Check the 64 bit library on any other Linux system.

     ```
     ls -l /usr/lib64/libstdc++.so.6
     ```

2. Check that the GCC compiler is at least at version 4.1.2

   Run the following command to display the version of GCC.

   ```
   gcc -v
   ```

3. Rebuild the application

   The commands to compile and link Linux C++ applications are described in Building 32-bit applications and Building 64-bit applications

What to do next

When you deploy your Linux C++ application, ensure that the same GCC runtime library is correctly installed on the run time system.

Migrating an application that uses the `MQRFH2` folder:

If you run an application that uses `MQRFH2` folders in an WebSphere MQ message, this migration task might be relevant to you. You might find, in a few cases, that the results of parsing an `MQRFH2` are different inVersion 7.5 to Version 6.0. Version 7.5 uses the `MQRFH2` folder to transport message properties. As a result, in some cases it reformats `MQRFH2` folders. The default setting of the `PROPCTL` queue and channel attribute eliminates most cases of incompatibility, but not all.

In many cases, IBM WebSphere MQ Version 6.0 did not parse the contents of the `MQRFH2`, and did not detect wrongly formatted `MQRFH2` headers. Applications could rely on receiving the content of an `MQRFH2` folder in the same format that it was created. In Version 7.5, messages that contain an `MQRFH2` folder are processed differently. Depending on the platform, and the folder contents, WebSphere MQ inspects, and in some cases modifies, the contents of an `MQRFH2` folder.

The reason for this change is that in Version 7.5, WebSphere MQ provides applications with programmatic access to the `MQRFH2` folder, using message properties. To support the message properties MQI in Version 7.5, WebSphere MQ parses, and modifies the contents of `MQRFH2` folders. As a result of extracting message properties from an `MQRFH2`, a number of `MQRFH2` properties can change between a message being sent and received:

1. The order of multiple MQRFH2 headers.
2. The order of folders in an MQRFH2.
3. The coded character set of an MQRFH2 folder
4. Attributes set on folders used for message properties are removed.
5. A single MQRFH2 might be broken into multiple chained MQRFH2 headers with different character sets.

The purpose of this task is twofold:

1. Configure WebSphere MQ to minimize the chance that application behavior changes as a result of the WebSphere MQ change.
2. Identify, and correct V6.0 programs that might be affected by the change in WebSphere MQ behavior.

You might have an application that relies on the format of an MQRFH2 remaining the same between the sender and receiver. The application worked in Version 6.0 but it no longer works in Version 7.5. Alternative settings of MQGMO_PROPERTIES and the queue and channel **PROPCTL** attributes provided in Version 7.0.1 do not fix the problem. A new **PROPCTL** queue attribute option in version Version 7.5, V6COMPAT, solves this migration problem for you. You must set the **PROPCTL**(V6COMPAT) option on both the sending and receiving queue, and on any additional intervening transmission queues. **PROPCTL**(V6COMPAT) preserves the format of MQRFH2 headers created by an application. The rules for setting the **PROPCTL**(V6COMPAT) option are different to other queue attributes; see PROPCTL.

Before you begin

The change affects an application connected to a Version 6.0 queue manager that exchanges messages with an application connected to a V7 queue manager. It also affects an application that you migrate from Version 6.0 to Version 7.5.

Many applications are unaffected, because they do not use the MQRFH2 folder explicitly. Even if they use the MQRFH2 folder explicitly, they are only affected if they place values directly into a property folder. For the list of defined property folders, see Defined property folder name and Ungrouped property folder name.

Tip:

Some MQRFH2 folders are created and used by WebSphere MQ, or by products such as WebSphere Message Broker, or WebSphere Application Server. WebSphere MQ creates MQRFH2 folders because an application uses a function such as clustering, JMS, or publish/subscribe. The folders created by WebSphere MQ do not require migration. Only if an application creates MQRFH2 folders itself, might you need to consider the tasks in this topic.

About this task

If channels and queues are created with the default setting for the **PROPCTL** attribute, most applications written for Version 6.0 continue to behave in the same way. If you select a different value for the **PROPCTL** attribute, you must consider the effect on a Version 6.0 application that uses the same channel or queue.

Tip: As a rule, if you do not modify any **PROPCTL** options, or set any MQGMO_PROPERTIES options, both old and new applications work as expected. Only if old and new applications use the same queues or channels, must you start thinking about whether to set MQGMO_PROPERTIES or **PROPCTL** options.

The tasks guide you in selecting the appropriate **PROPCTL** options, and reading and writing an MQRFH2 correctly.

Procedure

1. Decide whether the properties created in an application connected to a Version 7.5 queue manager must be forwarded to an application connected to a Version 6.0 queue manager.

   See "Using the **PROPCTL** channel property to control message properties."
2. Decide whether a Version 6.0 application, migrated to Version 7.5 without modification, is to receive message properties.

   See "Using the **PROPCTL** queue property to control message properties" on page 529.
3. Decide whether to set a MQGMO_PROPERTIES option in a new or modified application connected to a Version 7.5 queue manager.

   In many cases, you do not need to set an MQGMO_PROPERTIES option. The default option, MQGMO_PROPERTIES_AS_Q_DEF, is the best choice for existing applications. If you set a MQGMO_PROPERTIES option it overrides the setting of the **PROPCTL** queue attribute; see "Using the GMO_PROPERTIES_* option to control message properties" on page 530.
4. Design an application that parses properties in an MQRFH2 correctly. It will work when connected to a queue manager of any version.

   See "Parsing a message containing MQRFH2 headers correctly" on page 531.

   You can avoid this task by using the message property interface with new or changed Version 7.5 application

Using the **PROPCTL** *channel property to control message properties:*

The task guides you on setting the **PROPCTL** channel property for different types of channel. **PROPCTL** controls the transmission of message properties. **PROPCTL** can prevent an application connected to a V6 queue manager from receiving V7.0 message properties that it cannot process correctly.

When you create a channel to send messages from a V7.0 queue manager to a V6 queue manager, the channel has a default **PROPCTL** setting. The default setting of **PROPCTL** is COMPAT. The COMPAT option prevents new and unexpected properties being included in messages sent to the V6 queue manager. It also prevents new and unexpected properties being sent to earlier versions of queue managers. The attribute has no affect on messages sent between V7.0 queue managers. The precise effect of COMPAT is explained in Table 100 on page 730 in "**PROPCTL** channel options" on page 730.

Procedure

1. For sender or server channels, do one of the following options:
   a. Do nothing.

      The channel **PROPCTL** option defaults to COMPAT. With the COMPAT option set, applications that create no message properties interoperate between V7.0 queue managers. Applications that cause the queue manager to create message properties in V6[10], such as JMS or publish/subscribe applications, interoperate as well. New and changed applications that create new message properties in V7.0 generally do not interoperate with V6 without migration effort.

   b. Set **PROPCTL** to NONE or ALL, which are described in Table 100 on page 730 in "**PROPCTL** channel options" on page 730. Use these options if an application connected to a V6 queue manager interoperates with a V7.0 application that sets message properties. You can choose for the V6 application to receive none or all of the message properties.
2. For a cluster-sender channel, do nothing

   Cluster-sender channels default to COMPAT, and behave like sender or server channels. You cannot set **PROPCTL** on a cluster-sender channel. See "Property propagation in clusters containing both V6 and V7.0 queue managers " on page 529

[10] The property applies to channels connected to earlier versions of queue managers as well.

Property propagation in clusters containing both V6 and V7.0 queue managers

You might create a cluster of queue managers of different versions by migrating an existing cluster, queue manager by queue manager to V7.0. See "Migrating a queue manager cluster" on page 532. Another way of creating a mixed-version cluster is to add a V7.0 queue manager to an existing V6 cluster, or the other way around.

If a V6 queue manager is part of a cluster containing V7.0 queue managers, all V7.0 cluster-sender channels connect to it using **PROPCTL** set to COMPAT. You cannot change the **PROPCTL** setting on a V7.0 cluster-sender channel connected to a V6 or queue manager, or earlier.

To override this behavior you have two options:

1. Configure a point-to-point connection using non-cluster channels, and different queues.
2. Write a channel auto-definition exit to set the CLUSSDR **PROPCTL** attribute.

Using the **PROPCTL** *queue property to control message properties:*

Set the **PROPCTL** attribute on a queue to control which properties of a message are returned to an application that gets a message from the queue. Prevent an application migrated from V6 from receiving unexpected message properties in messages retrieved from the queue. Stage the conversion of applications to use the message property interface, instead of accessing an MQRFH2 header directly.

Before you begin

Setting the **PROPCTL** queue attribute does not affect all applications. It does not affect the following types of applications:

- JMS applications that use the WebSphere MQ classes for JMS to write messages to queues are *not* affected by the **PROPCTL** settings.

  JMS applications that use the WebSphere MQ classes for JMS to read messages from queues *are* affected by the **PROPCTL** settings.
- Applications that set an MQGMO_PROPERTIES_* option, unless they set it to MQGMO_PROPERTIES_AS_Q_DEF.

  If you set an MQGMO_PROPERTIES_* option to return message properties, query message properties by calling MQINQMP. You must create a message handle to use message properties.
- Applications that do not create MQRFH2 headers, and do not cause WebSphere MQ to create MQRFH2 headers.

  The main types of application that cause WebSphere MQ to create an MQRFH2 are JMS, publish/subscribe, and web services.

About this task

An MQI application can receive message properties from the same message on a Version 7.5 queue in different ways. Control the behavior of an unmodified application, such as one migrated from Version 6.0, by setting the **PROPCTL** queue attribute. Control the behavior of a new or modified application by setting the MQGMO_PROPERTIES_* options and using message handles. "Using the GMO_PROPERTIES_* option to control message properties" on page 530

Procedure

1. Identify the type of application that is to connect to the Version 7.5 queue manager. The application is one of four types:
   a. An application that open a queue with the default MQGMO_PROPERTIES option, MQGMO_PROPERTIES_AS_Q_DEF, or does not set the MQGMO_PROPERTIES option.

In most cases, no changes to the application are required. The application links dynamically to the new Version 7.5 MQI library. If the application is an MQI client, it can link to any version of the client libraries.

If the application behavior changes, you can change the **PROPCTL** queue attribute to correct the behavior of the application.

b. An application that opens a queue with a specific MQGMO message properties option, and gets message properties directly from MQRFH2 headers.

   If the application is an MQI client, it can be linked to any version of the client library.

c. An application that opens a queue with a specific MQGMO message properties option and creates a message handle. It can get message properties by calling MQINQMP

   If the application is an MQI client, it must link to at least the Version 7.0 version of the client library to use the message properties MQI calls.

d. A JMS application that uses the WebSphere MQ classes for JMS.

2. Select or create the queue on the Version 7.5 queue manager that the application is going to read messages from.
3. Set the **PROPCTL** queue attribute to one of the settings in Table 101 on page 732 in "**PROPCTL** queue options" on page 731. Or you can choose to leave it as the default, COMPAT.

   The effect of the **PROPCTL** queue attribute on an application depends on what type of application it is:

   An unmodified V6 application
   The property control settings in "**PROPCTL** queue options" on page 731 control which properties are returned to the application in MQRFH2 headers or can be retrieved by calling MQINQMP.

   An application that sets a non-default message property option, but does not use message handles.
   The application controls which properties are returned in MQRFH2 headers, unless the **PROPCTL** queue attribute is set to MQGMO_PROPERTIES_AS_Q_DEF.

   An application that sets a non-default message property option, and creates a message handles for the queue.
   All properties can be retrieved by calling MQINQMP, and are removed from MQRFH2 headers.

   A JMS application that uses the WebSphere MQ classes for JMS
   The application is unaffected by the **PROPCTL** queue attribute.

Related reference:

"MQGMO message property option settings" on page 734
Use MQGMO message property options to control how message properties are returned to an application.

Related information:

DEFINE queues

Using the GMO_PROPERTIES_* option to control message properties:

Set the GMO_PROPERTIES_* options in a new or changed MQI application connected to a Version 7.5 queue manager to control how properties are returned to the application.

Note: JMS applications do not use GMO_PROPERTIES_*. JMS and user properties are returned as JMS message properties. Any additional properties and folders in a JMS MQRFH2 are discarded when JMS reads a message as a JMS message. For a JMS application to read additional properties in a JMS MQRFH2, it must read the message as a MQ-style message. Set the destination property to WMQ_MESSAGE_BODY_MQ

Before you begin

GMO_PROPERTIES_* always overrides the setting of the queue attribute PROPCTL, unless the setting is MQGMO_PROPERTIES_AS_Q_DEF.

About this task

The procedure is laid out as a series of numbered design choices, one of which you must apply to your application. Following each choice is the value of the GMO_PROPERTIES option to choose to select the choice. Detailed descriptions about each option are to be found in Table 102 on page 735 in "MQGMO message property option settings" on page 734.

Procedure

The steps are expressed as alternative design choices.

1. Control the properties an existing application receives in an MQRFH2 by setting the **PROPCTL** queue attribute.
   - You can either set the **MQGMO** message property option to MQGMO_PROPERTIES_AS_Q_DEF or omit setting it. This option is the default.
2. Return all properties in the message handle, and cause an application that does not create a message handle to fail.
   - Set **MQGMO** Options to MQGMO_PROPERTIES_IN_HANDLE.
     a. If no handle is provided, the MQGET MQI call fails with reason code MQRC_HMSG_ERROR.
3. Remove all message properties, both those set by MQSETMP, and those written as message properties in a MQRFH2 header.
   - Set **MQGMO** Options to MQGMO_NO_PROPERTIES.

     If a handle is provided, no error is reported.
4. Process the entire MQRFH2 in the message without extracting any message properties, leaving them in the MQRFH2 folders.
   - Set **MQGMO** Options to MQGMO_PROPERTIES_FORCE_MQRFH2.
5. Do not alter the content of MQRFH2 headers that were returned to an application that was connected to a Version 6.0 queue manager and is now connected to a Version 7.5 queue manager.
   - Set **MQGMO** Options to MQGMO_PROPERTIES_COMPATIBILITY.

     Note: Sometimes, preserving the content of MQRFH2 headers is not sufficient to guarantee compatibility in the way MQRFH2 headers are returned. An application might not behave correctly when migrated from connecting to a Version 6.0 queue manager. In this case, set the **PROPCTL** V6COMPAT option on the sending and receiving queues.

Related reference:

"MQGMO message property option settings" on page 734
Use MQGMO message property options to control how message properties are returned to an application.

Related information:

DEFINE queues

Parsing a message containing MQRFH2 headers correctly:

Writing a procedure to parse MQRFH2 headers correctly is difficult. Use the syntax reference information and the hints in this topic to guide you. If at all possible, use the message property MQI and message properties rather than MQRFH2 folders and name/value pairs to transfer additional message properties. If you use the message properties interface, you remove the need to read, write, and parse an MQRFH2 header at all.

Before you begin

Consult the syntax description for both the fixed part, MQRFH2 Overview, and variable part, NameValueData (MQCHARn), of an MQRFH2.

About this task

The following procedure lists aspects of MQRFH2 syntax that are sometimes overlooked when creating or parsing MQRFH2 headers.

In many cases existing programs have not followed these rules exactly, but have worked satisfactorily when connected to IBM WebSphere MQ Version 6.0. When connected to Version 7.5, some of these applications behave differently, because the queue manager does modify the format of MQRFH2 headers. If this is a problem for you, setting the **PROPCTL** queue attribute, V6COMPAT restores the Version 6.0 behavior.

Procedure

- The length of *NameValueData* must be a multiple of 4. Some releases of WebSphere MQ tolerate other lengths.
- An MQRFH2 can be split into different headers. As long as the headers are chained, treat the header as one.
- Each MQRFH2 header can be encoded in a different character set.
- Each *NameValueData* folder can have a different *NameValueCSSID* value.

  NameValueData folders with different *NameValueCSSID* values must be placed in different MQRFH2 headers.
- A particular folder, ordinary or property folder, might be present more than once in one MQRFH2 header, or in multiple MQRFH2 headers.
- An element might be multiply defined. Only the first value of the element is significant. Having multiple values of an element is not an error.

  In the case of a few specific property folders, such as the mq. folder only the elements in the first folder are significant. In other cases, multiple definitions of a property is represented as a list of property values, not a single property value.
- Many variations on the syntax defined in NameValueData (MQCHARn) are tolerated by WebSphere MQ. Do not rely on WebSphere MQ detecting a badly formed MQRFH2.
  - It is not difficult to create an MQRFH2 with syntax that differs from the definition in NameValueData (MQCHARn). Some differences in syntax are detected, some are ignored and the data is retained, and some differences are ignored and the data is discarded. It is important to create and parse MQRFH2 headers correctly to avoid creating programs that work correctly today, but might not do so in the future.
  - You must test the behavior of applications using explicitly created MQRFH2 headers on each platform you run the application on. You must test the applications after each update of WebSphere MQ to a new release level. The reason for the extra testing is that the syntax of an MQRFH2 is only checked sufficient to that release and platform.
  - Use message properties, instead of creating and reading MQRFH2 headers explicitly, to avoid this testing burden. Using message properties transfers the burden of parsing an MQRFH2 correctly to the queue manager.

Migrating a queue manager cluster

Migrate a queue manager cluster by migrating each of the queue managers in the cluster. No changes since Version 6.0 specifically effect the migration of queue manager clusters. But you must consider what the effect is of migrating some queue managers in cluster, before all the queue managers are migrated.

Before you begin

Check that no cluster-specific migration issues are identified for the migration you are intending to perform. No cluster-specific changes have been made that affect migration between Version 6.0 and Version 7.1, and between Version 7.0.1 and Version 7.1.

Procedure

Consider the following issues that relate to migrating a queue manager cluster:

- Minimizing application outages.
- Measuring and verifying migration success and planning for backward migration if there are any migration problems.
- Taking advantage of new WebSphere MQ features.
- Managing the migration of a cluster in the context of the wider WebSphere MQ network and the systems architecture of your organization.

Steps

Some techniques for migrating a cluster with minimal planned and unplanned outages are described in the following topics:

Related concepts:

"How mixed version cluster repositories are updated" on page 362
Repositories store records for an object in a cluster in the version of the record format that matches the version of the queue manager hosting the repository. Repository queue managers forward object records, before they are stored, in the format that they are received in. The recipient ignores fields from a newer version, and uses default values for fields that are not present in the record.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

Related information:

Migrating WebSphere MQ queue manager clusters from WebSphere MQ V6 to V7

Migrating a queue manager cluster: Create a plan

Before carrying out the migration of a queue manager cluster, plan what you are going to do. Identify the roles that different queue managers play in the cluster, and decide in what order to migrate the queue managers.

Procedure

- What queue manager and application migration issues must be dealt with between the old and new versions?
- What system architecture and change control procedures must you consider?
- Consider migration questions specific to clusters, such as migrating full repositories first, and migrating overlapping clusters.
- Are any of the queue managers in a queue-sharing group, or part of a high-availability solution?
- Is the cluster a publish/subscribe cluster? Which queue manager is a cluster topic host?
- Decide whether to carry out a staged migration, or migrate all queue managers at the same time.
- Do you have a test system to migrate, and a production system?
- Document and test the plan before migrating production queue managers.

Related concepts:

"Application migration and interoperation" on page 360
WebSphere MQ supports running applications compiled and linked against previous versions of WebSphere MQ, with later levels of WebSphere MQ.

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

"How mixed version cluster repositories are updated" on page 362
Repositories store records for an object in a cluster in the version of the record format that matches the version of the queue manager hosting the repository. Repository queue managers forward object records, before they are stored, in the format that they are received in. The recipient ignores fields from a newer version, and uses default values for fields that are not present in the record.

"Queue manager cluster migration" on page 362
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Queue manager migration" on page 357
After upgrading an installation, queue manager migration might be required. Migration takes place when you start a queue manager.

"Queue-sharing group migration" on page 363
You can combine queue managers from different releases in a queue-sharing group. Limit the time you manage a mixed group to only as long as it takes to migrate all the queue managers to the same command level. You cannot combine a queue manager at Version 7.0 or later in the same queue-sharing group as queue managers earlier than Version 6.0. You must update all queue managers in a queue-sharing group with a coexistence PTF, before migrating any of them.

Related information:

Key roles for publish/subscribe cluster queue managers
Similar to point-to-point clusters, there are two key roles for queue managers in a publish/subscribe cluster; as full repository queue managers and as cluster topic hosts.

Migrating a queue manager cluster: Create a backout plan

Before performing a migration, decide on a backout plan in case of failure.

Before you begin

What backout capabilities do the queue managers in the cluster support? If no new function at the new level has been used, queue managers running on z/OS can be restored to an earlier level. On other platforms, the only backout option is to restore a queue manager to a previous state. In restoring a queue manager, you lose any persistent changes since the queue manager started running at the new level.

About this task

The backout plan must consider how to maintain the availability of the cluster. It must deal with any issues arising from migrating a queue manager in the cluster.

Procedure

The backout plan must describe the following points:

- What constitutes a successful migration.
- The conditions that trigger the backout procedure.
- Alternative backout actions, such as:
  1. Suspending a queue manager from the cluster.
  2. Backward migration
  3. Keeping a queue manager offline until an external issue is resolved.

Related concepts:

"Reverting a queue manager to a previous version" on page 358
On platforms other than z/OS it is not possible to reverse a migration. You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

Migrating a queue manager cluster: Migrating one cluster queue manager

Follow these steps to migrate a single queue manager in a cluster. Base your cluster migration plan on applying these steps to each queue manager in the cluster.

Procedure

1. Suspend the queue manager that you want to migrate from the cluster:
   a. Issue the **MQSC** command:
      ```
      SUSPEND QMGR CLUSTER(cluster name)
      ```
   b. Check that no messages are sent to the queue manager.

      You must close any application that continues to send messages to this queue manager. The cluster workload algorithm might choose the suspended queue manager. If there are no other valid destinations, or if an application has an affinity with the queue manager, it might select the queue manager.
2. Save a record of all cluster objects known by this queue manager. This data is used after migration to check that objects have been migrated successfully.
   a. Issue the command to view cluster queue managers.
      ```
      DISPLAY CLUSQMGR(*)
      ```
   b. Issue the command to view cluster queues.
      ```
      DISPLAY QC(*)
      ```
3. Save a record from the full repository of its view of the cluster objects owned by this queue manager. The record is used after migration to check that objects have been migrated successfully.
   a. Issue the command on the full repositories to display this queue manager.
      ```
      DISPLAY CLUSQMGR(migrated queue manager name)
      ```
   b. Issue the command on the full repositories to display the cluster queues for this queue manager
      ```
      DISPLAY QC(*) WHERE(CLUSQMGR EQ migrated queue manager name)
      ```
4. Migrate the queue manager.

   Do one of the queue manager migration tasks, depending on the platform; see "Migration tasks" on page 428.

   The queue manager migration process is, in outline:
   a. Stop the queue manager.
   b. Take a backup of the queue manager.
   c. Install the new version of WebSphere MQ.
   d. Restart the queue manager.
5. Ensure that all cluster objects have been migrated successfully.
   a. Issue the command to view cluster queue managers, and check the output against the data saved before migration.
      ```
      DISPLAY CLUSQMGR(*)
      ```
   b. Issue the command to view cluster queues and check the output against the data saved before migration.
      ```
      DISPLAY QC(*)
      ```
6. Check that the queue manager is communicating with the full repositories correctly.
7. Check that cluster channels to full repositories can start.

8. Check that the full repositories still have information about the migrated cluster queue manager and its cluster queues.
   a. Issue the command on the full repositories and check the output against the data saved before migration.

      ```
      DISPLAY CLUSQMGR(migrated_queue_manager_name)
      ```

   b. Issue the command on the full repositories and check the output against the data saved before migration.

      ```
      DISPLAY QC(*) WHERE(CLUSQMGR EQ migrated_queue_manager_name)
      ```

9. Test that applications on other queue managers can put messages to queues owned by the migrated cluster queue manager.
10. Test that applications on the migrated queue manager can put messages to the queues owned by other cluster queue managers.
11. Resume the queue manager.
    a. Issue `RESUME QMGR CLUSTER(`*cluster name*`)`
12. Closely monitor the queue manager and applications in the cluster for a while.

Related concepts:

"Queue manager migration" on page 357
After upgrading an installation, queue manager migration might be required. Migration takes place when you start a queue manager.

Related information:

DISPLAY CLUSQMGR

DISPLAY QUEUE

RESUME QMGR

SUSPEND QMGR

Migrating a queue manager cluster: Migrating the test system

Migrate each queue manager in the test system.

About this task

For each queue manager in the test system, in the order defined in the migration plan you developed in "Migrating a queue manager cluster: Create a plan" on page 533, migrate, and test the queue manager.

Migrating a queue manager cluster: Migrating the production system

Migrate each queue manager in the production system.

About this task

For each queue manager in the production system, in the order defined in the migration plan you developed in "Migrating a queue manager cluster: Create a plan" on page 533, migrate, and test the queue manager.

Windows: Migrating an MSCS configuration

Migrate queue managers in MSCS configuration one node at a time, following these instructions.

About this task

These steps are required for a rolling upgrade with a minimum amount of downtime. You must always upgrade an offline node with no online IBM WebSphere MQ resources. In an Active/Passive configuration, if the node is Passive, you must ensure it cannot be switched to Active during the upgrade process.

The example, "Migrating a four-node MSCS cluster from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5," shows this procedure applied to a four-node cluster.

Procedure

1. Modify the possible owners of the IBM WebSphere MQ resource to encompass only the Active node or nodes. With no owners assigned to Passive nodes, the IBM WebSphere MQ resource that is being migrated cannot be activated.
2. Ensure that the group containing the IBM WebSphere MQ resource is currently on one of the nodes defined as a possible owner. The group must include any applications connecting to the queue manager resource.
3. Stop the cluster service on the node being migrated. The MSCS cache is cleared of any IBM WebSphere MQ DLLs that have been registered.
4. Migrate the selected node by following the standard instructions in "Windows: Migrating a queue manager from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5" on page 509, "Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5" on page 474, or "Windows: Migrating a queue manager from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5" on page 445. Apply the required maintenance level.
5. Start the cluster service on the selected node.
6. On the next node to be migrated, ensure that the IBM WebSphere MQ resources are offline.
7. Remove this node from the list of possible owners. For clusters with more than two nodes, see the Additional considerations later in this topic.
8. Move the group containing the IBM WebSphere MQ resource to one of the possible owners and bring it online.
9. Repeat steps 3-8 as necessary for any remaining nodes.

Migrating a four-node MSCS cluster from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

The example in Table 63 on page 538 illustrates the steps involved in migrating a four-node MSCS cluster.

In the example IBM WebSphere MQ resources include queue managers, applications, and dependant MSCS resources, such as an IP address defined an as MSCS resource. In each step, the changes are italicized.

Step 1 Select the node to migrate and prepare it for upgrading from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5.

1. Select node 1 to be migrated and convert it into a Passive node with no running IBM WebSphere MQ resources.
2. Modify the possible owners of the group containing the IBM WebSphere MQ resources, to encompass only the required online nodes. Failover does not attempt to switch IBM WebSphere MQ resources to the node that is not a possible owner. It is safe to migrate that node.
3. Move the group containing the IBM WebSphere MQ resource to one of the nodes that is a possible owner, and bring it online.
4. Stop the cluster service on the node being migrated. Stopping the service clears the MSCS cache of any IBM WebSphere MQ libraries that have been registered for MSCS. The node goes offline.

Step 2 Migrate IBM WebSphere MQ from Version 6.0 to Version 7.5

Step 3 Start the cluster service on the selected node. The node becomes online, but it is not a possible owner, so no work is switched to it.

Step 4 Repeat steps 1 - 3 for node 2. Nodes 1 and 2 are now online, and you have migrated them to IBM WebSphere MQ Version 7.5. They are still doing no work, as they are not possible owners of any of the IBM WebSphere MQ resource groups.

Step 5 Migrate the cluster from running IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5. The number of migrated nodes is now greater or equal to the number of unmigrated nodes.

1. Change the set of possible owners from `3,4` to `1,2`.
2. Move the IBM WebSphere MQ resource groups from nodes 3 and 4 to nodes 1 and 2 and bring online.
3. From this point onward, the list of possible owners must include migrated nodes only. The WebSphere MQ resource must never failover to a node running a back level version of the product.

Note: If you must revert IBM WebSphere MQ to an earlier version, the IBM WebSphere MQ resources must be removed from MSCS control, before performing an uninstallation of IBM WebSphere MQ

Step 6 Migrate node 3 to Version 7.5.

1. Follow steps 1 - 3 for node 3.
2. Add node 3 to the list of possible owners.
3. Move the QMC resource group back from node 1 to node 3 and bring online again.

Step 7 Repeat step 6 for node 4.

Table 63. Migrating a four-node MSCS cluster

Steps		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Node 1	State	Online	*Offline*	Offline	*Online*	Online	Online	Online	Online
	Version	Version 6.0	Version 6.0	*Version 7.5*	Version 7.5	Version 7.5	Version 7.5	Version 7.5	Version 7.5
	Groups	QMA					*QMC, QMA*	*QMA*	QMA
Node 2	State	Online	Online	Online	Online	Online	Online	Online	Online
	Version	Version 6.0	Version 6.0	Version 6.0	Version 6.0	*Version 7.5*	Version 7.5	Version 7.5	Version 7.5
	Groups	QMB	QMB	QMB	QMB		*QMD, QMB*	QMD, QMB	*QMB*
Node 3	State	Online	Online	Online	Online	Online	Online	Online	Online
	Version	Version 6.0	Version 6.0	Version 6.0	Version 6.0	Version 6.0	Version 6.0	*Version 7.5*	Version 7.5
	Groups	QMC	QMC, QMA	QMC, QMA	QMC, QMA	QMC, QMA		*QMC*	QMC
Node 4	State	Online	Online	Online	Online	Online	Online	Online	Online
	Version	Version 6.0	Version 6.0	Version 6.0	Version 6.0	Version 6.0	Version 6.0	Version 6.0	*Version 7.5*
	Groups	QMD	QMD	QMD	QMD	QMD, QMB			*QMD*
Possible Owners		1,2,3,4	*2,3,4*	2,3,4	2,3,4	*3,4*	*1,2*	*1,2,3*	*1,2,3,4*
Task			Update 1			Update 2	Transfer	Update 3	Update 4

What to do next

Additional considerations in an MSCS setup with more than 2 nodes: A cluster might contain enough nodes for you to form a group of migrated queue managers and a group of unmigrated nodes. Switch to the migrated group when it contains half the number of queue managers. Before you have reached the half way point, the unmigrated group are possible owners. When you reach the half way point, switch the possible owners to the migrated group.

Related concepts:

"Migrate a queue manager in a high availability configuration" on page 365
Follow standard procedures to migrate a queue manager that is part of a high availability configuration on platforms other than z/OS.

Related reference:

"Windows: MSCS restriction with multiple installations" on page 594
When you install or upgrade to IBM WebSphere MQ Version 7.5, the first IBM WebSphere MQ installation on the server is the only one that can be used with Microsoft Cluster Server (MSCS). No other installations on the server can be used with MSCS. This restriction limits the use of MSCS with multiple IBM WebSphere MQ installations.

Migrating from a single instance to a multi-instance queue manager

To migrate a single instance queue manager to a multi-instance queue manager, you must move the queue manager data to a shared directory, and reconfigure the queue manager on two other servers.

Before you begin

You must check the prerequisites for running a multi-instance queue manager as part of this task. Some environments have been tested with multi-instance queue managers, and are known to work. They are AIX, Red Hat Linux, SUSE Linux Enterprise Server, HP-UX with the file system on Linux Red Hat, IBM i, and Windows Server. See Testing and support statement for WebSphere MQ multi-instance queue managers for the latest list of tested environments. The support statement has detailed version and prerequisite information for each environment it lists. Other environments might work; a test tool is provided with WebSphere MQ to assist you in qualifying other environments.

You must have three servers to run a multi-instance queue manager. One server has a shared file system to store the queue manager data and logs. The other servers run the active and standby instances of the queue manager.

About this task

You have a single-instance queue manager that you want to convert to a multi-instance queue manager. The queue manager conversion itself is straightforward, but you must do other tasks to create a fully automated production environment.

You must check the prerequisites for a multi-instance queue manager, set up the environment and check it. You must set up a monitoring and management system to detect if the multi-instance queue manager has failed and been automatically restarted. You can then find out what caused the restart, remedy it, and restart the standby. You must also modify applications, or the way applications are connected to the queue manager, so that they can resume processing after a queue manager restart.

Procedure

1. Check the operating system that you are going to run the queue manager on, and the file system on which the queue manager data and logs are stored on. Check that they can run a multi-instance queue manager.

a. Consult Testing and support statement for WebSphere MQ multi-instance queue managers. See whether the combination of operating system and file system is tested and capable of running a multi-instance queue manager.

   A shared file system must provide lease-based locking to be adequate to run multi-instance queue managers. Lease-based locking is a recent feature of some shared file systems, and in some case fixes are required. The support statement provides you with the essential information.

b. Run **amqmfsck** to verify that the file system is configured correctly.

   File systems are sometimes configured with performance at a premium over data integrity. It is important to check the file system configuration. A negative report from the **amqmfsck** tool tells you the settings are not adequate. A positive result is an indication that the file system is adequate, but the result is not a definitive statement that the file system is adequate. It is a good indication.

c. Run the integrity checking application provided in the technote, Testing a shared file system for compatibility with WebSphere MQ Multi-instance Queue Managers.

   The checking application tests that the queue manager is restarting correctly.

2. Configure a user and group to be able to access a share on the networked file system from each server that is running a queue manager instance.

   On Windows, the security IDs (SIDs) of the `mqm` group can be different; see Windows domains and multi-instance queue managers .

   On UNIX and Linux, the `uid` and `gid` for `mqm` in `/etc/password` must be the same on each system; see Create a multi-instance queue manager on Linux.

3. Set up a directory for the share on the networked file system with the correct access permissions.

   A typical configuration is to set up a single shared directory that contains all data and log directories for all queue managers that use the shared disk; see Share named qmgrs and log directories (version 7.0.1 onwards) in Example directory configurations on UNIX systems.

   For example, create a root directory on the share called `MQHA` that has subdirectories `data` and `logs`. Each queue manager creates its own data and log directories under `data` and `logs`. Create `MQHA` with the following properties:

   On Windows, create *drive*`\MQHA` on the shared drive. The owner is a member of `mqm`. `mqm` must have full-control authority. Create a share for *drive*`\MQHA`.

   On UNIX, create `/MQHA` on the shared drive. `/MQHA` is owned by the user and group `mqm` and has the access permissions `rwx`.

   If you are using an NFS v4 file server, add the line `/MQHA * rw,sync,no_wdelay,fsid=0`) to `etc/exports`, and then start the NFS daemon: `/etc/init.d/nfs start`.

4. Copy the queue manager data and the logs to the share.

   You can choose to copy files manually, by following the procedure to back up the queue manager. On Windows, you can run the **hamvmqm** command to move the queue manager data to the share. The **hamvmqm** command works for queue managers created before version 7.0.1, and not reconfigured with a datapath, or for queue managers that do not have a **DataPath** configuration attribute. Choose one of these methods:

   - Follow the instructions in Backing up queue manager data, copying the queue manager data to the share. You must use this method if the **DataPath** configuration attribute is specified for this queue manager.
   - Stop the queue manager, and then type the command,

   ```
   hamvmqm /m /dd share\data /dd share\logs
   ```

   Where *share* is to be the location of the data and logs that you created in step 3.

5. Update the queue manager configuration information stored on the current queue manager server.

   If you moved the queue manager data and logs by running the **hamvmqm** command, the command has already modified the configuration information correctly for you.

If you moved the queue manager data and logs manually, you must complete the following steps.

- On Windows:
  a. Modify the log registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeries\CurrentVersion\Configuration\QueueManager\QMgrName\Log
"LogPath"="share\\logs\\QMgrName\\"
```

  b. Modify the Prefix registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeries\CurrentVersion\Configuration\QueueManager\QMgrName
"Prefix"="share\\data"
```

- On UNIX, and Linux,
  a. Modify the `Log:` stanza in the queue manager `qm.ini` file, which is on the *share*:

```
LogPath=share/logs/QMgrName
```

  b. Modify the `QueueManager:` stanza in the WebSphere MQ `mqs.ini` file, which is typically in the `/var/mqm` directory on UNIX and Linux:

```
DataPath=share/data/QMgrName
```

Where, *QMgrName* is the representation of the queue manager name in the existing registry key on Windows. *QMgrName* is the `Directory` name in the `QueueManager:` stanza in the `mqs.ini` file on UNIX, and Linux. *share* is share where the data and logs are moved to.

6. Add the queue manager configuration information to the new queue manager server.
   a. Run the **dspmqinf** command to display the queue manager information

      Run the command on the server that ran the queue manager in version 6.0.

```
dspmqinf -o command QMgrName
```

   The command output is formatted ready to create a queue manager configuration.`addmqinf -s QueueManager -v Name=`*QMgrName* `-v Directory=`*QMgrName* `-v Prefix=d:\var\mqm Datapath=\`*share*`\`*data*`\`*QMgrName*

   b. Create a queue manager configuration on the other server.

      Run the **addmqinf** command copied from the previous output
7. Add the network address of the new server to the connection name in client and channel definitions.
   a. Find all the client, sender, and requester TCPIP settings that refer to the server.

      Client settings might be in Client Definition Tables (CCDT), in environment variables, in Java properties files, or in client code.

      Cluster channels automatically discover the connection name of a queue manager from its cluster receiver channel. As long as the cluster receiver channel name is blank or omitted, TCPIP discovers the IP address of the server hosting the queue manager.
   b. Modify the connection name for each of these connections to include the TCPIP addresses of both servers that are hosting the multi-instance queue manager.

      For example, change:

```
echo DISPLAY CHANNEL(ENGLAND) CONNAME | runmqsc QM1
```

```
5724-H72 (C) Copyright IBM Corp. 1994, 2009.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.
1: DISPLAY CHANNEL(ENGLAND) CONNAME
AMQ8414: Display Channel details.
CHANNEL(ENGLAND) CHLTYPE(SDR)
CONNAME(LONDON)
```

      Into:

```
echo ALTER CHANNEL(ENGLAND) CHLTYPE(SDR) CONNAME('LONDON, BRISTOL') | runmqsc QM1
```

8. Update your monitoring and management procedures to detect the queue manager restarting.
9. Update client applications to be automatically reconnectable, if appropriate.
10. Update the start procedure for your WebSphere MQ applications to be started as queue manager services.

11. Start each instance of the queue manager, permitting them to be highly available.

    The first instance of the queue manager that is started becomes the active instance.

    Issue the command twice, once on each server.

    ```
    strmqm -x QMgrName
    ```

What to do next

To get the highest availability out of multi-instance queue managers, you must design client applications to be reconnectable and server applications to be restartable; see Application recovery.

Related information:

amqmfsck (file system check)

Application recovery
Application recovery is the automated continuation of application processing after failover. Application recovery following failover requires careful design. Some applications need to be aware failover has taken place.

Automatic client reconnection
You can make your client applications reconnect automatically, without writing any additional code, by configuring a number of components.

Backing up queue manager data
Backing up queue manager data can help you to guard against possible loss of data caused by hardware errors.

Channel and client reconnection
Channel and client reconnection is an essential part of restoring message processing after a standby queue manager instance has become active.

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Create a multi-instance queue manager on Linux
An example shows how to set up a multi-instance queue manager on Linux. The setup is small to illustrate the concepts involved. The example is based on Linux Red Hat Enterprise 5. The steps differ on other UNIX platforms.

Moving a queue manager to MSCS storage
This procedure configures an existing queue manager to make it suitable for putting under MSCS control.

Multi-instance queue managers
Multi-instance queue managers are instances of the same queue manager configured on different servers. One instance of the queue manager is defined as the active instance and another instance is defined as the standby instance. If the active instance fails, the multi-instance queue manager restarts automatically on the standby server.

Queue manager configuration files, qm.ini
A queue manager configuration file, qm.ini, contains information relevant to a specific queue manager.

Shared file system
Shared files systems must provide data write integrity, guaranteed exclusive access to files and release locks on failure to work reliably with WebSphere MQ.

Testing a shared file system for compatibility with WebSphere MQ Multi-instance Queue Managers

Testing and support statement for WebSphere MQ multi-instance queue managers

The WebSphere MQ configuration file, mqs.ini
The IBM WebSphere MQ configuration file, mqs.ini, contains information relevant to all the queue managers on the node. It is created automatically during installation.

Verifying shared file system locking
Run **amqmfsck** to check whether a shared file system on UNIX systems meets the requirements for storing the queue manager data of a multi-instance queue manager. Run the IBM WebSphere MQ MQI client sample program **amqsfhac** in parallel with **amqmfsck** to demonstrate that a queue manager maintains message integrity during a failure.

Windows domains and multi-instance queue managers
A multi-instance queue manager on Windows requires its data and logs to be shared. The share must be accessible to all instances of the queue manager running on different servers or workstations. Configure the queue managers and share as part of a Windows domain. The queue manager can run on a domain workstation or server, or on the domain controller.

Working with services

Reverting to a single-instance queue manager

Revert a multi-instance queue manager to a single instance queue manager by stopping the standby instance. Then restart the active instance and do not set the flag that permits standby instances.

Before you begin

You have at least three servers configured to run a queue manager as a multi-instance queue manager. The queue manager is currently running as a multi-instance queue manager, with one standby instance active.

About this task

The task involves deactivating the active standby so that only the running multi-instance queue manager remains active. To prevent a standby instance being started in the future, you must stop the active instance and restart it. When you restart it, you start it as a single instance queue manager that prevents standby instances being started. The standby instance is stopped as a separate step, to give you the option of restarting the active instance at a later date. You can stop both instances by running the standard `endmqm` *QMgrName* command on the server running the active queue manager.

Procedure

1. Stop the standby queue manager instance.

   On the server running the standby instance:

   - Windows, UNIX, and Linux

     ```
     endmqm -w QMgrName
     ```

2. Stop the active queue manager instance.

   On the server running the active instance:

   - Windows, UNIX, and Linux

     ```
     endmqm -w (QMgrName)
     ```

3. Restart the queue manager, preventing standbys.

   On the server that is going to run the queue manager:

   - Windows, UNIX, and Linux

     ```
     strmqm QMgrName
     ```

What to do next

You might want to run the queue manager as a single instance on the same server as the queue manager data.

When the queue manager is stopped move the queue manager data back to the server that is running the queue manager. Alternatively install WebSphere MQ, and then move the queue manager configuration definition onto the server with the queue manager data. Both tasks are variations of steps in "Migrating from a single instance to a multi-instance queue manager" on page 539 to create a multi-instance queue manager.

Converting a publish/subscribe hierarchy to a publish/subscribe cluster

Conversion to a cluster requires manual steps and the whole hierarchy needs to be converted at the same time.

Before you begin

1. Migrate all the queue managers to the latest version, migrating the Version 6.0 publish/subscribe hierarchy; see "Migrating a IBM WebSphere MQ Version 6.0 publish/subscribe hierarchy to a IBM WebSphere MQ Version 7.5 hierarchy" on page 523
2. Advise users that the publish/subscribe system is not available while conversion takes place. The conversion takes time. All the queue managers in the hierarchy have to be converted together. Publications that are being processed are not lost, but all running publish/subscribe applications that use the hierarchy must be stopped. There are manual configuration tasks to be performed, some of which are not automatic transformations, before applications can be restarted.
3. To minimize disruption, plan, prepare, and test scripts to do the manual steps. Also review which topic objects are to be clustered with the publish/subscribe application owners.

About this task

The task is organized to minimize disruption. Migrate all the queue managers involved before starting the task. Step 1 on page 545 can be done in advance, without disrupting applications. Stop applications between steps 2 on page 545 to 5 on page 546 inclusive.

In step 5 on page 546, you need to define cluster topic objects on the cluster topic host. There are administrative advantages of using cluster topic objects, but that is not the reason for defining cluster topic objects during the conversion task. The principal reason for defining cluster topic objects as part of the conversion task to restore existing publish/subscribe applications to a fully working state.

Existing publish/subscribe applications that use a hierarchy do not work in a cluster, unless you identify and define the correct cluster topics. In hierarchical publish/subscribe, subscriptions are propagated to every queue manager in the hierarchy, as long as the subscription is registered with **SUBSCOPE**(ALL). In clustered publish/subscribe no subscriptions are propagated to the cluster, unless the subscription resolves to a topic with the attribute **CLUSTER**(*clusterName*) set, and **SUBSCOPE**(ALL).

Part of the migration procedure is to identify or create topic objects that are to be given the cluster attribute. Application knowledge is required to choose the correct topics to associate with cluster topic objects. Here is some guidance to help you in choosing which topics to cluster.

- If your Version 6.0 publish/subscribe applications used streams, part of the migration process is to create topic objects corresponding to the streams. These topic objects are prime candidates to be made into cluster topic objects.

  The migration process creates one topic object for each stream that is defined on a Version 6.0 queue manager. Thus, depending on what streams are defined on what queue managers, you might find different topic objects created on different queue managers.

  Unlike any other clustered topic object, if you are doing queued publish/subscribe you must define the topic objects that correspond to streams on every queue manager in the cluster. Do *not* follow the typical procedure, and define the topic object only on the cluster topic host. If you are doing integrated publish/subscribe, you only need to define the cluster topic once. The queued publish/subscribe daemon requires a locally defined topic object corresponding to the name of the stream in SYSTEM.QPUBSUB.QUEUE.NAMELIST. You must set the **CLUSTER** attribute on each of the automatically defined local topic objects. The cluster attribute causes subscriptions to topics that resolve to a stream topic object to be propagated to other members of the cluster. The attributes of the topic object are resolved to the cluster topic object that is locally defined, and not to the latest cluster topic object to be defined.

- Identify or create as few topic objects to be clustered as the application design requires. Allow other topics to inherit from these topic objects. To this end, look for topics near the root of the topic trees of your publish/subscribe applications, and make them clustered.
- Identify any Version 6.0 publish/subscribe applications that are going to be hard to migrate from a hierarchy to a cluster.

  Applications that use the default stream and have a flat topic space, or do not have topics you can clearly identify as the root of their topic trees, are hard to migrate. It might be difficult to know how to define cluster topic objects associated with the topics an application uses. Do *not*, except as a last resort, set the cluster attribute on `SYSTEM.BASE.TOPIC` as a way of causing all topics to inherit the cluster attribute. It might be worthwhile to convert your existing Version 6.0 publish/subscribe applications that use the default stream, to use named streams. Then each stream converts to a defined topic object that you can cluster.

  Note: If you set the cluster attribute on `SYSTEM.BASE.TOPIC`, you do not need to set it on any other topics that inherit from `SYSTEM.BASE.TOPIC`.

Procedure

1. Create a cluster and add all the queue managers in the hierarchy to the cluster.
   a. Create a cluster or nominate an existing cluster.

      The cluster does not need to be an existing publish/subscribe cluster.

      Use IBM WebSphere MQ script commands (MQSC), or any other type of administration command or utility that is available on your platform, such as the IBM WebSphere MQ Explorer.

   b. Check that each queue manager is in the cluster.

      Run the MQSC command:

      ```
      DISPLAY CLUSQMGR(*)
      ```

      If a queue manager is not in the cluster, add it.
2. Stop publish/subscribe applications.

   Allow current work to complete.

   a. Stop all publish/subscribe publishers.

      No more work must be put into the system.

      Do not stop new work by disabling the input stream queues - the input streams are needed to process publications that remain in the hierarchy. You must stop the applications themselves.

   b. Switch off queued publish/subscribe on all the queue managers in the hierarchy.

      Leave version 7.0.1 integrated publish/subscribe running.

      Run the following MQSC command on all the queue managers in the hierarchy.

      ```
      ALTER QMGR PSMODE(COMPAT)
      ```

   c. Wait.

      Transmission queues and channels used to connect queue managers in the hierarchy must complete processing publications that are already in the system.

      When no more publications are left in transmission queues and channels, all the publications have reached their destination queue manager. The queue manager delivers the publications to subscribers when queued publish/subscribe is re-enabled.
3. Delete all the channels and transmission queues used to connect queue managers in the hierarchy.
   a. Stop all the channels used to connect the queue managers in the hierarchy.

      Run the following MQSC command on all the queue managers in the hierarchy.

```
STOP CHANNEL(SenderChanName) MODE(QUIESCE)
```

b. Delete all the channels used to connect the queue managers in the hierarchy.

   Run the following MQSC command on all the queue managers in the hierarchy.

```
DELETE CHANNEL(SenderChanName)
DELETE CHANNEL(ReceiverChanName)
```

c. Delete the transmission queues associated with the channels that were deleted.

   Run the following MQSC command on all the queue managers in the hierarchy.

```
DELETE QLOCAL(xmitQName)
```

4. Delete the queue manager hierarchy.
   a. Enable queued publish/subscribe on each queue manager in the hierarchy using the MQSC command:

```
ALTER QMGR PSMODE(ENABLE)
```

   b. Run the following MQSC command on each queue manager in the hierarchy, except the uppermost parent in the hierarchy which has no parent.

```
ALTER QMGR PARENT(' ')
```

   Alternatively, on IBM i run the following commands to remove queue managers from the hierarchy.

   1) Run **WRKMQMPS PUBSUBNAME**(*parentQmgrName*) to display the hierarchy.
   2) Use **option 4=Remove** to remove the parent from the hierarchy.
   3) Use **option 34=Work with Pub/Sub** to move down the subhierarchy
   4) Repeat options **4** and **34** until there are no child queue managers displayed.

   c. Before proceeding to the next step confirm that all the hierarchical relationships have been canceled.

   Run the following MQSC command on each queue manager.

```
DISPLAY PUBSUB TYPE(ALL)
```

5. Set the **CLUSTER** attribute on the topic objects that you have decided to make clustered.
   a. If you need to create cluster topic objects, define them on the cluster topic host.

      For integrated publish/subscribe, define cluster topic objects on only one queue manager.

   b. If you are setting the cluster attribute on existing topic objects, created by **strmqbrk**, set the cluster attribute on the topic object defined on the cluster topic host. Delete the topic object from the other queue managers.

      Multiple definitions of a cluster topic object on multiple queue managers in a cluster can lead to problems.

   c. Review whether to delete any topic objects that were created by **strmqbrk** that are not clustered.

      Subscriptions that inherit from these topic objects do not get propagated to other queue managers in the cluster.

6. Restart publish/subscribe applications.

Related information:

DISPLAY CLUSQMGR

DISPLAY CLUSQMGR

WebSphere MQ cluster commands

WebSphere Event Broker and WebSphere Message Broker migration tasks

Migrate the publish/subscribe broker and collectives from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1 to Version 7.5. Migrate SCADA applications that use IBM WebSphere MQ Telemetry.

Before you begin

IBM WebSphere MQ does not support the content-based filtering provided in WebSphere Event Broker version 6.0. If you use this function and intend to continue to do so, you must migrate to WebSphere Message Broker version 7.0. Content-based filtering is supported in IBM WebSphere MQ Version 7.5, and used by WebSphere Message Broker; see Enabling content-based filtering with publish/subscribe.

About this task

Users can migrate applications from WebSphere Event Broker version 6.0 to use the publish/subscribe broker in IBM WebSphere MQ Version 7.5.

If you are upgrading from WebSphere Message Broker version 6.0 or 6.1, you must migrate any applications that use publish/subscribe. You must also migrate message flow applications that use SCADAInput or SCADAOutput nodes.

Procedure

If WebSphere Event Broker or WebSphere Message Broker are connected into a collective you must do the task "Migrating a WebSphere Message or Event Broker publish/subscribe collective to a WebSphere MQ publish/subscribe cluster" on page 550 before "Migrating publish/subscribe configuration data from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1" on page 548.

- Migrate publish/subscribe collectives to publish/subscribe clusters.

  "Migrating a WebSphere Message or Event Broker publish/subscribe collective to a WebSphere MQ publish/subscribe cluster" on page 550
- Migrate the publish/subscribe broker from WebSphere Event Broker or WebSphere Message Broker to IBM WebSphere MQ.

  "Migrating publish/subscribe configuration data from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1" on page 548
- Migrate SCADA applications to use IBM WebSphere MQ Telemetry.

  "Migration of telemetry applications from using WebSphere Message Broker version 6 to use IBM WebSphere MQ Telemetry and WebSphere Message Broker version 7.0" on page 553

What to do next

The WebSphere Event Broker or WebSphere Message Broker publish/subscribe information is not deleted by the migration process. This information is stored in the subscription database tables, and is available to use until you explicitly delete it.

Related concepts:

"Migration of the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker" on page 369
The publish/subscribe broker used in WebSphere Message Broker version 6.0 and 6.1 is replaced by the integrated publish/subscribe broker in IBM WebSphere MQ. IBM WebSphere MQ replaces WebSphere Event Broker version 6.0. The migration to IBM WebSphere MQ has two parts. The migration of publish/subscribe configuration data, and conversion of publish/subscribe collectives to publish/subscribe clusters.

"Migration phases during the migration of WebSphere Message Broker or WebSphere Event Broker" on page 370
The migration of the publish/subscribe broker from WebSphere Message Broker or WebSphere Event Broker to WebSphere MQ has three phases. The rehearsal phase, the initial phase, and the completion phase.

Related reference:

"Differences between the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker version 6, and the publish/subscribe broker in WebSphere MQ version 7" on page 669
WebSphere Event Broker version 6 is superseded by the publish/subscribe broker in WebSphere MQ version 7.0.1. The internal publish/subscribe broker in WebSphere Message Broker version 6.0 and 6.1 is replaced by WebSphere MQ version 7.0.1 publish/subscribe in WebSphere Message Broker version 7. Applications are affected by a number of changes in behavior by the migration to the publish/subscribe broker in WebSphere MQ.

Related information:

Enabling content-based filtering with publish/subscribe

Selecting on the content of a message

Migrating publish/subscribe configuration data from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1

Complete these tasks to migrate publish/subscribe configuration data from WebSphere Event Broker Version 6.0 to WebSphere MQ Version 7.0.1 and later versions.

Before you begin

Familiarize yourself with the information in "Migration of the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker" on page 369 and "Differences between the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker version 6, and the publish/subscribe broker in WebSphere MQ version 7" on page 669.

Install WebSphere MQ Version 7.0.1 or a later version.

On distributed systems, set up and initialize a command environment for the WebSphere Event Broker from which you are migrating. You are going to run the WebSphere MQ commands and WebSphere Event Broker using this environment.

Set the queue manager **PSMODE** attribute to COMPAT, using the following command: ALTER QMGR PSMODE(COMPAT). In COMPAT mode the queue manager processes publish/subscribe calls. It does not process any publish/subscribe command messages.

Procedure

1. Optional: Run the rehearsal phase of the migration.

   For example, on supported platforms other than z/OS, use the following command to rehearse the migration from a broker named BRK1:

   ```
   migmbbrk -r -b BRK1
   ```

   On z/OS, run the equivalent **CSQUMGMB** utility.
2. Review the contents of the log file and the security commands file to check what would happen in a real migration.
3. Run the initial phase of the migration. For example, on supported platforms other than z/OS, use the following command to run the initial phase from a broker named BRK1:

   ```
   migmbbrk -t -b BRK1
   ```

   On z/OS, run the equivalent **CSQUMGMB** utility.
4. Review the commands in the security commands file amqmigrateacl.txt. Ensure that they create a security environment that is equivalent to your broker security environment. The migration rehearsal might find an Access Control List (ACL) entry that denies access. If so, it reports it in the security command file and advises that the ACL migration must be performed manually. There are two alternative ways to migrate the access control information:
   - Modify the security commands, then go to step 5 on page 549

     Modify the generated commands so that they set up a security environment in the queue manager that is equivalent to the security environment that existed in the broker.

- Revise the security definitions in the broker to generate security commands that require no modification.

  Modify the security environment in the broker to be similar to WebSphere MQ by following these steps:

  a. Set the ACL entry for the root of the topic tree to **Deny** for both publish and subscribe.

     The **Deny** setting matches the setting for the root of the WebSphere MQ topic tree. **Deny** is the exact opposite of the broker default. It can result in many publishers and subscribers no longer having permission to perform operations that they have been doing successfully.

     The next two steps restore the successful operation of the denied publishers and subscribers:

     1) Remove all ACL entries that deny access, apart from the entry in step 4a.

        Many of these entries are likely to be redundant following denying access at the root of the topic tree. Other entries might require more extensive changes.

     2) Add ACL entries, where required, to grant access and to restore a correct security environment.

  b. Run the initial phase of the migration again and review the security command file.

     If your changes are correct, it creates a security environment in the queue manager that is equivalent to the security environment that existed in the broker.

5. Run the security commands to set up the security environment before you run the completion phase of the migration.

   If you do not run the security commands again, the migration fails.

6. Run the completion phase of the migration.

   - On supported platforms other than z/OS, run the following command to migrate the publish/subscribe configuration data from broker BRK1

     ```
     migmbbrk -c -o -b BRK1
     ```

     The command overwrites subscriptions or retained publications that exist in the queue manager and have the same name as a migrating subscription or retained publication.

   - On z/OS, use the equivalent **CSQUMGMB** utility.

   The completion phase migrates the publish/subscribe configuration data to the queue manager and shuts down the broker. It creates a log file and a security commands file.

   Note: It is possible that the broker state has changed since the initial phase was run and that additional topic objects are now required. If so, the completion phase creates these topic objects as necessary. The completion phase does not delete topic objects that have become unnecessary. You must delete topic objects that you do not require.

7. Add the name of every queue that is named in an WebSphere Event Broker or WebSphere Message Broker MQInput node to the namelist SYSTEM.QPUBSUB.QUEUE.NAMELIST.

   To edit a namelist, use either the WebSphere MQ Explorer or the following MQSC command:

   ```
   ALTER NAMELIST
   ```

   Note: Each queue name referenced in the SYSTEM.QPUBSUB.QUEUE.NAMELIST namelist also has an associated Topic object. You must define the Topic objects before adding the associated queue name to the SYSTEM.QPUBSUB.QUEUE.NAMELIST namelist. For more information about setting up the SYSTEM.QPUBSUB.QUEUE.NAMELIST namelist, see Mapping between streams and topics.

   WebSphere Event Broker and WebSphere Message Broker put published messages to, and read published messages from, queues using an MQInput node. By listing these queues in SYSTEM.QPUBSUB.QUEUE.NAMELIST, the queue manager monitors these queues for published messages.

   When the migration shuts down the broker, published messages are not read from the MQInput node in this way any more. Instead, the queue manager monitors the relevant queues.

8. Check the broker domain Event Log to confirm that the broker has been shut down.

9. Use the following command to set the **PSMODE** queue manager attribute to ENABLED.

```
ALTER QMGR PSMODE(ENABLED)
```

PSMODE(ENABLED) starts the queued publish/subscribe command message interface, and the integrated publish/subscribe call interface. The queue manager does all publish/subscribe processing.

What to do next

The migration process uses a queue called SYSTEM.TEMP.MIGMBBRK.REPLY.QUEUE to receive messages from the broker. When the process starts it checks for the existence of the queue and creates or clears it as necessary. When it has finished, it attempts to delete the queue. However, because the broker also uses the queue to send replies, te broker might still have the queue open. If so, the migration process is unable to delete the queue. When you have completed the migration, check if SYSTEM.TEMP.MIGMBBRK.REPLY.QUEUE exists, and if it does, delete it.

Related concepts:

"Publish/Subscribe Broker: Access Control List (ACL) migration" on page 371
An automated process migrates the publish/subscribe broker from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ. The process does not migrate the ACLs automatically. The migration task provides two methods to migrate the ACL to WebSphere MQ.

Related reference:

"Publish/Subscribe Broker: ACL migration on Linux and UNIX platforms" on page 669
On Linux and UNIX platforms, it is possible for a username and groupname to have the same identifier. In which case the migration process is unable to determine whether an ACL applies to the username, the groupname, or both.

"Publish/Subscribe Broker: ACL wildcard subscriptions" on page 670
A subscription can use a wildcard scheme to subscribe to multiple topics. Access to one or more of the resulting matched topics might be denied by its access control list. The result in WebSphere MQ differs after migration from the result in WebSphere Event Broker and WebSphere Message Broker before migration.

"Publish/Subscribe Broker: Extra topic objects created by migration" on page 671
Administrative topic objects are created by the migration process. Some of the topic objects might be unnecessary. Although they do not cause any difference in behavior, deleting unwanted topic objects makes publish/subscribe easier to administer.

Related information:

Migrating publish/subscribe information from WebSphere MQ Version 7.0.1 to WebSphere Message Broker Version 6.0 or 6.1

IBM Redbook: Connecting Your Business Using IBM WebSphere Message Broker V7 as an ESB

Migrating a WebSphere Message or Event Broker publish/subscribe collective to a WebSphere MQ publish/subscribe cluster

Complete this task to migrate a publish/subscribe collective. Migrate the collective from WebSphere Event Broker Version 6.0 or WebSphere Message Broker Version 6.0 or 6.1. Migrate it to a WebSphere MQ publish/subscribe cluster.

Before you begin

Install WebSphere MQ version 7.0.1 or a later version.

Make sure that you are familiar with the information about migration from WebSphere Message Broker and WebSphere Event Broker. See "Differences between the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker version 6, and the publish/subscribe broker in WebSphere MQ version 7" on page 669 and "Publish/Subscribe Broker: Access Control List (ACL) migration" on page 371.

Make sure that you are familiar with the first tasks to set up a new cluster and add a queue manager to it. See Configuring a queue manager cluster

On distributed systems, set up and initialize a command environment in which WebSphere MQ commands and WebSphere Event Broker commands can run.

Set the queue manager **PSMODE** attribute to COMPAT, using the following command: ALTER QMGR PSMODE(COMPAT)

Procedure

1. Stop all the publish/subscribe applications in the topology so that all in-flight messages are processed.
2. Use the **MQSC** command DISPLAY QUEUE(*) to check that the current depth on the following queues is zero:
   - SYSTEM.BROKER.INTERBROKER.QUEUE.1A
   - SYSTEM.BROKER.INTERBROKER.QUEUE.1N
   - SYSTEM.BROKER.INTERBROKER.QUEUE.1T
   - SYSTEM.BROKER.CONTROL.QUEUE
   - Transmit queues
   - Message flow input queues
3. Upgrade the underlying queue manager of each broker in the collective to WebSphere MQ version 7.0.1 or a later version.
4. If any queue manager is currently connected using a manually defined channel and transmit queue, delete the manually defined channel now. During the removal of the transmit queues it is possible that messages from the WebSphere Message Broker configuration manager are left stranded on transmit queues. The configuration messages have not been processed off the queue, because all applications are stopped. The stranded messages are not a problem. If the configuration manager does not receive a timely response it automatically attempts to reestablish communication. When migration is complete, the configuration manager is no longer used.

   Note: If you roll back the publish/subscribe configuration migration, you must recreate these manually defined channels and transmit queues. If you do not, the cluster channels are used for communication between the brokers.
5. Use **MQSC** commands to set up a queue manager cluster containing all the queue managers that are associated with the brokers:
   a. Decide on a cluster name
   b. Nominate two queue managers as full repository queue managers:
   ```
   ALTER QMGR REPOS('clusname')
   ```
   c. Define the cluster receiver channel on each queue manager:
   ```
   DEFINE CHANNEL('to.qmgr_name') CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('(hostname(fr_listener_port)') CLUSTER('cl
   ```
   d. Start the channel:
   ```
   START CHANNEL('to.qmgr_name')
   ```
   e. Define the cluster to send to the full repository on each queue manager:
   ```
   DEFINE CHANNEL('to.fr_qmgr_name') CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('fr hostname(fr_listener_port)') CLUSTER
   ```
   f. Start the channel:
   ```
   START CHANNEL('to.fr_qmgr_name')
   ```
6. Create the publish/subscribe cluster. On each queue manager, alter the topic SYSTEM.BASE.TOPIC to add it to the cluster:
   ```
   ALTER TOPIC(SYSTEM.BASE.TOPIC) CLUSTER('clusname')
   ```
7. Wait for the cluster topics to propagate and then check that all queue managers are correctly participating in the cluster:
   ```
   DISPLAY CLUSQMGR(*)
   ```

8. For each queue manager, migrate the publish/subscribe configuration data to WebSphere MQ. See "Migrating publish/subscribe configuration data from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1" on page 548.
9. After migrating the publish/subscribe configuration data for each broker to each queue manager, resynchronize proxy subscriptions. The proxy subscriptions are resynchronized with all the other queue managers in the publish/subscribe cluster. Run this MQSC command to trigger the resynchronization:

   ```
   REFRESH QMGR TYPE (PROXYSUB)
   ```

   Note: If you roll back and rerun the broker publish/subscribe migration, you must run this resynchronization step and all the steps that follow it.
10. Wait for all proxy subscriptions and retained publications to be propagated. Check that the status of subscriptions and publications is what you expect to see:
    a. Check the proxy subscriptions:

    ```
    DISPLAY SUB(*) SUBTYPE(PROXY)
    ```

    b. Check retained publications:

    ```
    DISPLAY TPSTATUS('#') RETAINED
    ```

    c. Use this MQSC command to check that the current depth of the following queues is zero:

    ```
    DISPLAY QLOCAL(*)

    SYSTEM.INTER.QMGR.CONTROL
    SYSTEM.INTER.QMGR.PUBS
    SYSTEM.INTER.QMGR.FANREQ
    SYSTEM.CLUSTER.TRANSMIT.QUEUE
    ```
11. Restart your publish/subscribe applications.

What to do next

The migration process uses a queue called `SYSTEM.TEMP.MIGMBBRK.REPLY.QUEUE` to receive messages from the broker. When the process starts it checks for the existence of the queue and creates or clears it as necessary. When it has finished, it attempts to delete the queue. However, because the broker also uses the queue to send replies, te broker might still have the queue open. If so, the migration process is unable to delete the queue. When you have completed the migration, check if `SYSTEM.TEMP.MIGMBBRK.REPLY.QUEUE` exists, and if it does, delete it.

Related concepts:

"Publish/Subscribe Broker: Access Control List (ACL) migration" on page 371
An automated process migrates the publish/subscribe broker from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ. The process does not migrate the ACLs automatically. The migration task provides two methods to migrate the ACL to WebSphere MQ.

Related tasks:

"Migrating publish/subscribe configuration data from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1" on page 548
Complete these tasks to migrate publish/subscribe configuration data from WebSphere Event Broker Version 6.0 to WebSphere MQ Version 7.0.1 and later versions.

Related reference:

"Differences between the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker version 6, and the publish/subscribe broker in WebSphere MQ version 7" on page 669
WebSphere Event Broker version 6 is superseded by the publish/subscribe broker in WebSphere MQ version 7.0.1. The internal publish/subscribe broker in WebSphere Message Broker version 6.0 and 6.1 is replaced by WebSphere MQ version 7.0.1 publish/subscribe in WebSphere Message Broker version 7. Applications are affected by a number of changes in behavior by the migration to the publish/subscribe broker in WebSphere MQ.

Related information:

➡ Migrating publish/subscribe collectives from WebSphere Message Broker Version 6.0 or 6.1 to WebSphere MQ Version 7.0.1

Configuring a queue manager cluster
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

Migration of telemetry applications from using WebSphere Message Broker version 6 to use IBM WebSphere MQ Telemetry and WebSphere Message Broker version 7.0

Migrating telemetry applications from using WebSphere Message Broker v6 to use IBM WebSphere MQ Telemetry and WebSphere Message Broker v7 requires a number of steps. Only the last step involves modifying the telemetry application. The first steps are migrating IBM WebSphere MQ and WebSphere Message Broker to v7.

You can run WebSphere Message Broker v6 with IBM WebSphere MQ Telemetry, if you convert all your SCADA flows to use IBM WebSphere MQ Telemetry. You must migrate IBM WebSphere MQ to Version 7.1 to use IBM WebSphere MQ Telemetry.

Before you begin

Your target environment is going to run the IBM WebSphere MQ Telemetry feature on IBM WebSphere MQ Version 7.1, with WebSphere Message Broker v7. IBM WebSphere MQ Telemetry runs on Windows, AIX, and Linux, see http://www.ibm.com/software/products/en/wmq-telemetry for detailed information.

Consider setting up parallel development and test environments before starting the task of upgrading and migrating all the components. In a large production environment, migrating the telemetry applications is the last step after upgrading IBM WebSphere MQ and WebSphere Message Broker to v7.

An alternative migration path is to migrate telemetry applications before migrating WebSphere Message Broker to v7. It is likely to be more effort to follow this path. New functionality in WebSphere Message Broker v7, namely republishing to the same topic, makes it easier to migrate some flows from SCADA to IBM WebSphere MQ Telemetry. By migrating the telemetry applications last you also avoid testing telemetry twice. First, you would test telemetry with WebSphere Message Broker v6, using the publish/subscribe engine in the broker. Second, you would test telemetry with WebSphere Message Broker v7, using IBM WebSphere MQ as the publish/subscribe engine.

About this task

You have created telemetry applications that use the MQTT protocol and the Supervisory Control And Data Acquisition (SCADA) nodes in WebSphere Message Broker v6. You want to upgrade your WebSphere Message Broker deployment from v6 to v7, but SCADA nodes are not supported in v7 of WebSphere Message Broker. Connectivity support for the MQTT protocol has been moved from WebSphere Message Broker to IBM WebSphere MQ. You want to leave as much of the telemetry application unchanged as you can. You must migrate the telemetry connectivity between the MQTT client and WebSphere Message Broker. Migrate it from the SCADA nodes in WebSphere Message Broker to the telemetry channels in IBM WebSphere MQ Version 7.1.

Procedure

1. Upgrade IBM WebSphere MQ

   See “Planning IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5 migration by platform” on page 418.
2. Install the IBM WebSphere MQ Telemetry feature.

   See WebSphere MQ Telemetry installation.

3. Upgrade WebSphere Message Broker to v7.

   See Upgrade to WebSphere Message Broker v7.

4. Modify existing MQTT client applications to use v3 of the MQTT protocol.

   If your client applications are already using v3 of the MQTT protocol, no modifications to the applications are required.

   If you are using an MQTT client library with your applications, the library might not support MQTT v3. Migrate the library to use one of the C or Java MQTT v3 libraries provided with IBM WebSphere MQ Telemetry .

5. Modify your message flows from using SCADA nodes to using MQInput or JMSInput nodes.

   See Migration from SCADA node of WebSphere Message Broker to WebSphere MQ Telemetry.

Related information:

WebSphere MQ Telemetry

Telemetry concepts and scenarios for monitoring and control

Installing WebSphere MQ Telemetry
From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

Administering WebSphere MQ Telemetry

Developing applications for WebSphere MQ Telemetry

WebSphere MQ Telemetry Reference

Troubleshooting for WebSphere MQ Telemetry

List of changes by version, release, and maintenance level

Any changes in the behavior of WebSphere MQ that affect existing applications or configurations are listed in the following topics. The changes are listed by migration path. For each migration path, all changes are listed. Study the changes to identify what migration tasks you must perform.

Changes are listed for currently supported migration paths. To migrate a queue manager from a release that is not listed, see “Migration paths” on page 320.

Related concepts:

“Migration paths” on page 320
Where can you find topics describing how to migrate WebSphere MQ from one release to another? Can you revert a migration to an earlier release? Can you migrate from an unsupported WebSphere MQ release to any later release directly?

Related tasks:

“Migration tasks” on page 428
Follow these tasks to migrate queue managers from one version to another, and to migrate queue managers that are in clusters. You will also find assistance in migrating publish/subscribe or telemetry from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ.

Coexistence

Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

The coexistence section lists restrictions in the use of objects and commands when they are used with queue managers at multiple command levels. The queue managers might be running on a single server, or in a cluster.

Related concepts:

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks:

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

JMS: Administered objects

Version 6.0 JMS administered objects, such as connection factories and destinations are compatible with later releases.

JMS applications can use connection factory or destination objects created in IBM WebSphere MQ Version 6.0. Any new properties, that did not exist in Version 6.0, assume their default value.

IBM WebSphere MQ Version 6.0 JMS applications can use connection factory or destination objects created in later releases. Any of the new properties that do not exist in Version 6.0 are ignored.

Cluster workload balancing in a mixed cluster

A cluster can contain queue managers running at IBM WebSphere MQ Version 7.5, and any currently supported level of IBM WebSphere MQ, but new features cannot be exploited from queue managers at an earlier level however.

As IBM WebSphere MQ Version 7.5 adds a new **DEFBIND** value called GROUP to queues, applications on queue managers earlier than Version 7.5 must not open or put messages to queues specifying the new value. When an application ignores this limitation, the workload balancing behaviour (for example: BIND_ON_OPEN or BIND_NOT_FIXED) is undefined.

z/OS: ISPF operations and control panels

When using the operations and control panels, the WebSphere MQ libraries you use in ISPF must be compatible with the queue manager you are working with.

Table 64 on page 556 shows which versions of the operations and controls panels you use in ISPF are compatible with which levels of queue manager. Version 7.1 panels are incompatible with any release before Version 6.0.

Table 64. Compatibility of queue manager versions with operations and control panel versions

Version	Queue-sharing group containing a mixture of Version 6.0, Version 7.0.1, and Version 7.1 queue managers.	Version 7.1 queue manager	Version 7.0.1 queue manager	Version 7.0 queue manager	Version 6.0 queue manager
Version 7.1 panel	Compatible with restrictions and warnings	Compatible	Compatible with restrictions and warnings		
Version 7.0.1 panel	Compatible with warnings. [See note]		Compatible	Compatible with restrictions and warnings	
Version 7.0 panel	Compatible with warnings. [See note]	Not compatible	Compatible	Compatible	Compatible with restrictions and warnings
Version 6.0 panel	Compatible with warnings. [See note]				Compatible

Note: The migration and coexistence PTF must be applied to the WebSphere MQ libraries that are concatenated with the ISPF application. If it is not applied, the panels do not work.

z/OS: Queue sharing group coexistence

A queue-sharing group can contain queue managers running on IBM WebSphere MQ Version 6.0, and on later releases. The queue managers can access the same shared queues and other shared objects. A Version 6.0 queue-manager must have the coexistence PTF applied for the latest release. You cannot combine a queue manager in a queue-sharing group from a version earlier than Version 6.0 with a queue manager from a later release.

Note: After the coexistence PTF has been applied, the Version 6.0 queue manager must be started at least once. The first time it is started it migrates IBM WebSphere MQ internal objects.

You must migrate a Version 5.3.1 queue-sharing group to a Version 6.0 or later queue-sharing group. If you do not, you cannot start a queue manager on a version later than Version 6.0 as a member of the group. Once you have migrated the queue-sharing group to a version later than Version 6.0, you cannot start a queue manager from a version earlier than Version 6.0 as a member of the group.

You cannot start all the queue managers in a queue-sharing group containing queue managers that are at Version 6.0 and later versions, and queue managers from versions earlier than Version 6.0. If an earlier version queue manager is active in the group you cannot start a Version 6.0 or later version queue manager in the group. Similarly, if a Version 6.0 or later version queue manager is active in the group, you cannot start an earlier version queue manager in the group. You cannot start a Version 6.0 or later version queue manager in a migrated queue-sharing group unless the coexistence PTF for the latest version queue manager in the group is applied to every earlier version queue manager in the group.

Only run queue managers in a mixed-version queue-sharing group for the time it takes to migrate all queue managers to the later version. If the queue-sharing group contains queue managers with a mixture of versions, IBM WebSphere MQ for z/OS running on the later version allows prototyping with the new facilities on the queue manager, and tolerates operation at the Version 6.0 level or above.

z/OS: Coupling Facility restrictions in a mixed queue-sharing group

The migration of queue managers in a queue-sharing group, and coupling facilities, to a new level must be coordinated. Until all the queue managers in a group are at the new level, you are restricted in actions you can perform on the coupling facility.

The following two restrictions apply to a coupling facility structure object in a queue-sharing group that contains queue managers at an earlier version than 6.0. When all the queue managers in the queue-sharing group have been migrated to version 6.0 or later, and started, the restrictions no longer apply.

1. You cannot alter a coupling facility structure object from **CFLEVEL**(3) to **CFLEVEL**(4).
2. You cannot delete a coupling facility structure object.

When you have updated the coupling facility structure object to **CFLEVEL**(4), queue managers at a version earlier than 6.0 cannot connect to it. They cannot access the queues defined on it, or the messages stored on the queues.

z/OS: Properties of objects in a mixed queue-sharing group

Attributes that did not exist in earlier versions can be created and altered on queue managers of a later version in a mixed queue-sharing group. The attributes are not available to queue managers in the group that are at an earlier level.

You can define and alter objects with **QSGDISP**(GROUP) from a queue manager at Version 7.0.1 or later. The objects, and their resulting copy objects, are accessible on all the queue managers. On queue managers at Version 6.0 new attributes and values are not available.

You can define a queue alias object with the properties **QSGDISP**(GROUP) and **TARGTYPE**(TOPIC). The **TARGTYPE** property is not recognized by a Version 6.0 queue manager. The copy object of the queue alias is interpreted on the Version 6.0 queue manager as referencing a queue rather than the intended topic.

z/OS: MQSC commands in a mixed queue-sharing group

Existing **MQSC** commands using new keywords and attribute values can be entered for routing to a migrated queue manager. You can enter the commands on any queue manager. Route the commands using **CMDSCOPE**. Commands with new keywords and attribute values, or new commands, routed to a previous version of queue manager, fail.

z/OS: PCF commands in a mixed queue-sharing group

On z/OS, create PCF commands only on a version 6.0 or higher queue manager. Also, if a command uses the parameter MQCACF_COMMAND_SCOPE to define a target queue manager, the target queue manager must be version 6 or later

z/OS: Mixed case RACF in a queue-sharing group

How you turn on mixed-case RACF support varies according to whether you have activated queue manager level checking or queue sharing group level checking, or both.

Mixed-case RACF security is supported in WebSphere MQ Version 7.0. Ensure that you have installed a level of the security product that supports mixed case security, that you have applied any updates required by WebSphere MQ, and that the new WebSphere MQ RACF classes are installed and active. Because a queue-sharing group can contain a mix of WebSphere MQ Version 7.0 queue managers and WebSphere MQ Version 6.0 queue managers, you must consider the following situations:

- “Mixed-case RACF support with queue manager level checking active” on page 558
- “Mixed-case RACF support with queue sharing group level checking active” on page 558
- “Mixed-case RACF support with queue manager and queue sharing group level checking active” on page 558

Mixed-case RACF support with queue manager level checking active

There are multiple queue managers in a queue-sharing group running on different versions of WebSphere MQ, and they all have queue manager only security checking active.

The process for turning on mixed-case RACF support for those queue managers that are running on WebSphere MQ Version 7.0 with mixed-case security is the same as when the queue managers are not in a queue-sharing group. Follow the migration steps in Migration to mixed-case security.

Mixed-case RACF support with queue sharing group level checking active

Multiple queue managers in a queue-sharing group are running on different versions of WebSphere MQ, and they all have queue-sharing group only security checking active.

Carry out the following procedure if you have a shared mixed case resource, protected through a mixed case profile, but with some queue managers using `SCYCASE(UPPER)`.

To turn on mixed-case RACF support for those queue managers that are running on WebSphere MQ Version 7.0 with mixed-case security, see Migration to mixed-case security:

1. Carry out Steps 1-3.
2. Carry out Steps 4-6 for each queue manager that you are changing.
3. Carry out Step 7 for all queue managers in the queue-sharing group.
4. Carry out Step 8, but with the following additional consideration:

   If you are defining a shared mixed-case resource, you can protect it with a mixed-case profile in a mixed-case class on those queue managers that are using that support. On queue managers that are not using that support, the resource can only be protected using a generic uppercase profile in an uppercase class.

   In summary, queue managers using mixed-case support perform security checks against the mixed-case classes and queue managers using uppercase support (whatever the release of the product) perform security checks against the uppercase classes.

Mixed-case RACF support with queue manager and queue sharing group level checking active

There are multiple queue managers in a queue-sharing group running on different versions of WebSphere MQ, and they all have queue manager and queue-sharing group security checking active.

To turn on mixed-case RACF support for those queue managers that are running on WebSphere MQ Version 7.0 with mixed-case security, see “Mixed-case RACF support with queue sharing group level checking active.”

Changes from IBM WebSphere MQ Version 7.1 to IBM WebSphere MQ Version 7.5

All IBM WebSphere MQ changes that affect the migration of a queue manager from Version 7.1 to Version 7.5 are listed.

New capabilities that do not affect existing IBM WebSphere MQ applications are not listed.

Changes in Version 7.5

Table 65. Summary of Version 7.5 changes

What changed	Link to more information
New channel state: Switching	Display channel and cluster status: Switching
Command level	Command level changed to 750
endmqm command	Change of behavior of the endmqm command
Data types that have a new current version in Version 7.5	Changes to data types
▶V7.5.0.8 JMS ExceptionListener (Version 7.5.0, Fix Pack 8)	JMS: Exception listener changes in Version 7.5

Stabilizations, deprecations and removals in Version 7.5

Table 66. Summary of stabilizations, deprecations and removals at Version 7.5

Stabilized, deprecated or removed feature	Link to more information
dspmqsver command (WebSphere MQ Advanced Message Security)	Removal of dspmqsver command

Display channel and cluster status: Switching

From IBM WebSphere MQ Version 7.5, a cluster-sender channel that is switching its configuration to a different cluster transmission queue has a new channel state: Switching.

When migrating from a release before Version 7.5 to Version 7.5 or later, existing application programs are not affected by the new state.

System management programs that monitor channel or cluster status might receive the new state as a result of a inquiry.

The state is set during the short interval while the channel modifies the destination transmission queue that messages are stored on. Before the switching state is set, messages are stored on the previously associated transmission queue. After the switching state, messages are stored on the newly configured transmission queue. The channel enters the switching state if a cluster-sender channel is starting, a configuration change is required, and the conditions for starting the switch are met.

Related information:

Channel states
A channel can be in one of many states at any time. Some states also have substates. From a given state a channel can move into other states.

Cluster transmission queues and cluster-sender channels
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

DISPLAY CHSTATUS

Inquire Channel Status (Response)

Inquire Cluster Queue Manager (Response)

MQCHS_* (Command format Channel Status)

Command level changed to 750

The command level on platforms other than z/OS and IBM i changes to 750 in Version 7.5. z/OS and IBM i are at command level 710.

Related information:

MQCMDL_* (Command Levels)

CommandLevel (MQLONG)

Change in behavior of the `endmqm` command

Issuing an **endmqm** command and **dspmq** command immediately after each other might return misleading status.

When issuing an **endmqm -c** or **endmqm -w** command, in the unlikely event that a dspmq command is issued in the small timeframe between the applications disconnecting and the queue manager actually stopping, the dspmq command might report the status as `Ending immediately`, even though a controlled shutdown is actually happening.

Related information:

endmqm

dspmq

Removal of `dspmqsver` command

Before IBM WebSphere MQ, the **dspmqsver** command was used to display the version of IBM WebSphere MQ Advanced Message Security.

From IBM WebSphere MQ Version 7.5, IBM WebSphere MQ Advanced Message Security is a component of the main product and the version information is displayed as part of the **dspmqver** command.

JMS: Exception listener changes in Version 7.5

In IBM WebSphere MQ Version 7.5, when using the IBM WebSphere MQ classes for JMS, changes are made to the way in which an application's JMS ExceptionListener is invoked.

Exception listener changes in Version 7.5

The following changes only apply to JMS applications that register MessageListener objects to consume messages asynchronously.

APAR IT14820, included from IBM WebSphere MQ Version 7.5.0, Fix Pack 8, fixed a defect where an application's JMS ExceptionListener would not be invoked for non-connection broken exceptions (for example MQRC_GET_INHIBITED) even though the ASYNC_EXCEPTIONS property on the JMS Connection Factory used by the application, was set to ASYNC_EXCEPTIONS_ALL. This was the default value prior to Version 7.5.0, Fix Pack 8.

From Version 7.5.0, Fix Pack 8, to maintain behavior for current JMS applications that configure a JMS MessageListener and a JMS ExceptionListener, and to ensure that the IBM WebSphere MQ classes for JMS are consistent with the JMS specification, the default value for the ASYNC_EXCEPTIONS JMS ConnectionFactory property has been changed to ASYNC_EXCEPTIONS_CONNECTIONBROKEN for the IBM WebSphere MQ classes for JMS. As a result, by default, only exceptions corresponding to broken connection error codes are delivered to an application's JMS ExceptionListener.

From Version 7.5.0, Fix Pack 8, the IBM WebSphere MQ classes for JMS have also been updated such that JMSExceptions relating to non-connection broken errors, which occur during message delivery to asynchronous message consumers, are still delivered to a registered ExceptionListener when the JMS ConnectionFactory used by the application has the ASYNC_EXCEPTIONS property set to the value ASYNC_EXCEPTIONS_ALL.

Exception listeners changes in Version 7.0

In later releases of the product, JMS exception listeners behave differently than they did in Version 6.0. Applications might receive more or fewer exceptions than they did in Version 6.0.

In Version 6.0 of IBM WebSphere MQ, an exception listener was called to inform the application of any error condition that occurred asynchronously to the application execution. During the processing of a message for an asynchronous consumer, the application might have had no other means to discover the exception. The errors that resulted in the exception listener being called included connection broken events, or if an attempt was made to process an unreadable message.

Previously, with Version 6.0 JMS clients, a property had to be explicitly set if connection broken type exceptions that occurred during synchronous API calls were to be passed to the application's ExceptionListener code. However, from Version 7.0, connection broken exceptions are expected to be passed, without the need to set any properties, to an application's ExceptionListener regardless of whether they were making synchronous API calls or receiving messages asynchronously to a MessageListener.

From Version 7.0, you can set the **ASYNC_EXCEPTIONS** property on JMS Connection Factory objects using the method:

```
setIntProperty(JmsConstants.ASYNC_EXCEPTIONS, int)
```

.

V7.5.0.8 Before APAR IT14820, included from IBM WebSphere MQ Version 7.5.0, Fix Pack 8, the default value of this property was `ASYNC_EXCEPTIONS_ALL`. With APAR IT14820, the default value is changed to `ASYNC_EXCEPTIONS_CONNECTIONBROKEN`. For more information, see “JMS: Exception listener changes in Version 7.5” on page 560.

If you set the **ASYNC_EXCEPTIONS** property to `ASYNC_EXCEPTIONS_ALL`, the exception listener is called for all broken connection exceptions. The exception listener is also called for all exceptions that occur outside the scope of a synchronous JMS API call. Creating a JMS Connection with an ExceptionListener from a JMS Connection Factory that has **ASYNC_EXCEPTIONS** set to `ASYNC_EXCEPTIONS_ALL` should provide broadly the same behavior as a Version 6.0 JMS client that had the `activateExceptionListener` Java system property set to the value true.

If you set the **ASYNC_EXCEPTIONS** property to `ASYNC_EXCEPTIONS_CONNECTIONBROKEN`, only exceptions indicating a broken connection are sent to the exception listener. These exceptions include connection broken exceptions occurring both synchronously and asynchronously. They do not include any other asynchronous errors such as for unreadable messages. In this mode, if the exception listener is triggered, the connection can be considered to have failed. It is no longer possible to use the connection to send or receive messages.

Applications must take appropriate action, such as attempting to remake the connection, when exception listener calls are made.

Related information:

Exceptions in WebSphere MQ classes for JMS

ASYNCEXCEPTION

Changes from IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5

All IBM WebSphere MQ changes that affect the migration of a queue manager from Version 7.0.1 to Version 7.5 are listed.

New capabilities that do not affect existing IBM WebSphere MQ applications are not listed.

On platforms other than z/OS you cannot reverse queue manager migration to remove the effect of changes.

New features and their supported APIs for Version 7.0.1

Use this information to learn about the WebSphere MQ Version 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

The following topics contain tables that show supported WebSphere MQ Version 7.0.1 APIs. Use the information in these tables in conjunction with the information in the system requirements web pages. For links to system requirements information for all releases of WebSphere MQ, see the System Requirements web page.

New features and their supported APIs for Version 7.0.1: JMS:

Use this information to learn about the WebSphere MQ Version 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

JMS

The following table shows which features are supported for JMS.

	JMS										
	Distributed platforms				**z/OS**						
API	**Basic**[1]	**WAS and other Java EE**[2]	**Other TM**	**Db2 SP**[3]	**Basic**	**WAS**[2]	**Other Java EE**	**Db2 SP**[3]	**Other TM**	**CICS®**	**IMS™**
V7 Publish Subscribe	Y	Y	Y	N/A[4]	Y	Y	N/A	N/A[4]		N/A	N/A
Browse with Mark	N	N	N	N/A[4]	N	N	N/A	N/A[4]		N/A	N/A
Message Properties	Y[5]	Y[5]	Y[5]	N/A[4]	Y[5]	Y[5]	N/A	N/A[4]	Y[5]	N/A	N/A
Message Selectors	Y	Y	Y	N/A[4]			N/A	N/A[4]		N/A	N/A
Asynchronous consume	Y	Y	Y	N/A[4]			N/A	N/A[4]		N/A	N/A
Content filtering Publish/ Subscribe	N	N	N	N/A[4]	N	N	N/A	N/A[4]	N	N/A	N/A
Group/ Segment messages	N	N	N	N/A[4]	N	N	N/A	N/A[4]	N	N/A	N/A
PCF Classes	Y[6]	Y[6]	Y[6]	N/A[4]	Y[6]	Y[6]	N/A	N/A[4]	Y[6]	N/A	N/A
Global TX Participant	Y[7]	Y	Y[7]	N/A[4]	Y[8]	Y	N/A	N/A[4]	Y[8]	N/A	N/A
Global TX Coordinator	N	N	N	N/A[4]	N	N	N/A	N/A	N	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Db2 SP Distributed: cannot work reliably because of the restricted Java environment.
5. Cannot see the whole MQI namespace, therefore an MQI application might generate messages whose properties cannot be read.
6. Can build/parse messages but not use the MessageAgent classes & methods.
7. Open source JTA coordinators might not be supported.
8. RRS is the coordinator.

	JMS										
	Distributed platforms				**z/OS**						
Client communication (not running on top of C client)	**Basic[1]**	**WAS and other Java EE[2]**	**Other TM**	**Db2 SP[3]**	**Basic**	**WAS[2]**	**Other JavaEE**	**Db2 SP[3]**	**Other TM**	**CICS**	**IMS**
Read ahead	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Asynchronous put	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Pre-connect exit	N	N	N	N/A	N/A	N	N/A	N/A	N/A	N/A	N/A
CCDT	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Multiple CONNAME	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Auto reconnect	Y	Y[5]	Y	N/A	N/A	Y[5]	N/A	N/A	N/A	N/A	N/A
Channel compression	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Exit chaining	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
MQCSP/ MQXR_SEC_PARMS	Y[4]	Y[4]	Y[4]	N/A	N/A	Y[4]	N/A	N/A	N/A	N/A	N/A
Native SSL	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Not consistent across all clients.
5. Use RA features for reconnect, not the FAP-level operation.

Related information:

WebSphere MQ resource adapter V7.1 and later statement of support

Running IBM WebSphere MQ classes for Java applications within Java Platform Enterprise Edition

New features and their supported APIs for Version 7.0.1: Java:

Use this information to learn about the WebSphere MQ Version 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

Java

The following table shows which features are supported for Java.

	Base Java											
	Distributed platforms					**z/OS**						
API	**Basic**[1]	**WAS**[2]	**Other JEE**	**Other TM**	**Db2 SP**[3]	**Basic**	**WAS**[2]	**Other JEE**	**Db2 SP**[3]	**Other TM**	**CICS**	**IMS**
V7 Publish Subscribe	Y	N/A	N/A[4]	N/A	N/A[6]	Y	N/A	N/A	N/A[6]	N/A	Y	N/A
Browse with Mark	Y	N/A	N/A[4]	N/A	N/A[6]	Y	N/A	N/A	N/A[6]	N/A	Y	N/A
Message Properties	N	N/A	N/A[4]	N/A	N/A[6]	N	N/A	N/A	N/A[6]	N/A	N	N/A
Message Selectors	Y	N/A	N/A[4]	N/A	N/A[6]	Y	N/A	N/A	N/A[6]	N/A	Y	N/A
Asynchronous consume	N	N/A	N/A[4]	N/A	N/A[6]	N	N/A	N/A	N/A[6]	N/A	N	N/A
Content filtering Publish/ Subscribe	N	N/A	N/A[4]	N/A	N/A[6]	N	N/A	N/A	N/A[6]	N/A	N	N/A
Group/ Segment messages	Y	N/A	N/A[4]	N/A	N/A[6]	Y	N/A	N/A	N/A[6]	N/A	Y	N/A
PCF Classes	Y	N/A	N/A[4]	N/A	N/A[6]	Y	N/A	N/A	N/A[6]	N/A	Y	N/A
Global TX Participant	N	N/A	N/A[4]	N/A	N/A[6]	N	N/A	N/A	N/A[6]	N/A	N	N/A
Global TX Coordinator	Y	N/A	N/A[4]	N/A	N/A[6]	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. While it might work (some customers have used this environment), it is not recommended or fully supported.

5. RRS is the coordinator.
6. Db2 SP on Distributed platform cannot work reliably because of the restricted Java environment.

	Base Java											
	Distributed platforms					z/OS						
Client communication (not running on top of C client)	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]	Basic	WAS[2]	Other JEE	Db2 SP[3]	Other TM	CICS	IMS
Read ahead	Y	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
CCDT	Y	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Channel compression	Y	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Exit chaining	Y	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
MQCSP / MQXR_SEC_PARMS	Y[4]	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A
Native SSL	Y	N/A	N/A	N/A	N/A	N/A[5]	N/A	N/A	N/A	N/A	N/A	N/A

Note:
1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Not consistent across all clients.
5. Java client not supported in any environment on z/OS

Related information:

WebSphere MQ resource adapter V7.1 and later statement of support

Running IBM WebSphere MQ classes for Java applications within Java Platform Enterprise Edition

New features and their supported APIs for Version 7.0.1: XMS .NET:

Use this information to learn about the WebSphere MQ Version 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

XMS .NET

The following table shows which features are supported for XMS .NET.

	XMS .NET				
	Distributed platforms				
API	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
V7 Publish Subscribe	Y	N/A	N/A	N/A	N/A
Browse with Mark	N	N/A	N/A	N/A	N/A
Message Properties	Y[4]	N/A	N/A	N/A	N/A
Message Selectors	Y	N/A	N/A	N/A	N/A
Asynchronous consume	Y	N/A	N/A	N/A	N/A
Content filtering Publish/Subscribe	N	N/A	N/A	N/A	N/A
Group/Segment messages	N	N/A	N/A	N/A	N/A
PCF Classes	N	N/A	N/A	N/A	N/A
Global TX Participant	Y[5]	N/A	N/A	N/A	N/A
Global TX Coordinator	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Cannot see the whole MQI namespace, therefore an MQI application might generate messages whose properties cannot be read.
5. Works with Microsoft DTC only.

	XMS .NET				
	Distributed				
Client communication (not running on top of C client)	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
Read ahead	Y	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A
CCDT	Y	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A

	XMS .NET				
	Distributed				
Client communication (not running on top of C client)	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
Channel compression	N	N/A	N/A	N/A	N/A
Exit chaining	N	N/A	N/A	N/A	N/A
MQCSP/ MQXR_SEC_PARMS	N	N/A	N/A	N/A	N/A
Native SSL	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.

New features and their supported APIs for Version 7.0.1: XMS C & C++:

Use this information to learn about the WebSphere MQ Version 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

XMS C & C++

The following table shows which features are supported for XMS C & C++.

	XMS C & C++				
	Distributed				
API	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
V7 Publish Subscribe	Y	N/A	N/A	N/A	N/A
Browse with Mark	N	N/A	N/A	N/A	N/A
Message Properties	Y	N/A	N/A	N/A	N/A
Message Selectors	Y	N/A	N/A	N/A	N/A
Asynchronous consume	Y	N/A	N/A	N/A	N/A
Content filtering Publish/Subscribe	N	N/A	N/A	N/A	N/A
Group/Segment messages	N	N/A	N/A	N/A	N/A
PCF Classes	N	N/A	N/A	N/A	N/A
Global TX Participant	N	N/A	N/A	N/A	N/A
Global TX Coordinator	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.

	XMS C & C++				
	Distributed				
Client communication (not running on top of C client)	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
Read ahead	Y	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A
CCDT	N	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A
Channel compression	Y	N/A	N/A	N/A	N/A
Exit chaining	Y	N/A	N/A	N/A	N/A
MQCSP/ MQXR_SEC_PARMS	N/A	N/A	N/A	N/A	N/A
Native SSL	Y	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.

New features and their supported APIs for Version 7.0.1: .NET:

Use this information to learn about the WebSphere MQ Version 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

.NET

The following table shows which features are supported for .NET.

	.NET				
	Distributed				
API	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
V7 Publish Subscribe	Y	N/A	N/A	N/A	N/A
Browse with Mark	Y	N/A	N/A	N/A	N/A
Message Properties	Y[4]	N/A	N/A	N/A	N/A
Message Selectors	N	N/A	N/A	N/A	N/A
Asynchronous consume	N	N/A	N/A	N/A	N/A
Content filtering Publish/Subscribe	N	N/A	N/A	N/A	N/A
Group/Segment messages	N	N/A	N/A	N/A	N/A
PCF Classes	N	N/A	N/A	N/A	N/A
Global TX Participant	Y[5]	N/A	N/A	N/A	N/A
Global TX Coordinator	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Cannot see the whole MQI namespace, therefore an MQI application might generate messages whose properties cannot be read.
5. Works with Microsoft DTC only.

	.NET				
	Distributed				
Client communication (not running on top of C client)	Basic[1]	WAS[2]	Other JEE	Other TM	Db2 SP[3]
Read ahead	Y	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A
CCDT	Y	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A
Channel compression	N	N/A	N/A	N/A	N/A
Exit chaining	N	N/A	N/A	N/A	N/A

	.NET				
	Distributed				
Client communication (not running on top of C client)	**Basic**[1]	**WAS**[2]	**Other JEE**	**Other TM**	**Db2 SP**[3]
MQCSP/ MQXR_SEC_PARMS	N	N/A	N/A	N/A	N/A
Native SSL	N[4]	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Support available in Unmanaged mode only.

Channel authentication

When you migrate a queue manager to Version 7.5, channel authentication using channel authentication records is disabled. Channels continue to work as before. If you create a queue manager in Version 7.5, channel authentication using channel authentication records is enabled, but with minimal additional checking. Some channels might fail to start.

Migrated queue managers

Channel authentication is disabled for migrated queue managers.

To start using channel authentication records you must run this MQSC command:

```
ALTER QMGR CHLAUTH(ENABLED)
```

New queue managers

Channel authentication is enabled for new queue managers.

You want to connect existing queue managers or IBM WebSphere MQ MQI client applications to a newly created queue manager. Most connections work without specifying any channel authentication records. The following exceptions are to prevent privileged access to the queue manager, and access to system channels.

1. Privileged user IDs asserted by a client-connection channel are blocked by means of the special value `*MQADMIN`.

   ```
   SET CHLAUTH('*') TYPE(BLOCKUSER) USERLIST('*MQADMIN') +
   DESCR('Default rule to disallow privileged users')
   ```

2. Except for the channel used by IBM WebSphere MQ Explorer, all `SYSTEM.*` channels are blocked.

   ```
   SET CHLAUTH('SYSTEM.*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) +
   DESCR('Default rule to disable all SYSTEM channels')

   SET CHLAUTH(SYSTEM.ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(CHANNEL) +
   DESCR('Default rule to allow MQ Explorer access')
   ```

Note: This behaviour is default for all new IBM WebSphere MQ Version 7.5 queue managers on startup.

If you must work around the exceptions, you can run an MQSC command to add in more rules to allow channels blocked by the default rules to connect, or disable channel authentication checking:

```
ALTER QMGR CHLAUTH(DISABLED)
```

Related information:

Channel authentication records

Change in behavior of the `endmqm` command

Issuing an **`endmqm`** command and **`dspmq`** command immediately after each other might return misleading status.

When issuing an **`endmqm -c`** or **`endmqm -w`** command, in the unlikely event that a dspmq command is issued in the small timeframe between the applications disconnecting and the queue manager actually stopping, the dspmq command might report the status as `Ending immediately`, even though a controlled shutdown is actually happening.

Related information:

endmqm

dspmq

Changes to cluster error recovery on servers other than z/OS

Before Version 7.5, if a queue manager detected a problem with the local repository manager managing a cluster, it updated the error log. In some cases, it then stopped managing clusters. The queue manager continued to exchange applications messages with a cluster, relying on its increasingly out of date cache of cluster definitions. From Version 7.5 onwards, the queue manager reruns operations that caused problems, until the problems are resolved. If, after five days, the problems are not resolved, the queue manager shuts down to prevent the cache becoming more out of date. As the cache becomes more out of date, it causes a greater number of problems. The changed behavior regarding cluster errors in Version 7.5 does not apply to z/OS.

Every aspect of cluster management is handled for a queue manager by the local repository manager process, `amqrrmfa`. The process runs on all queue managers, even if there are no cluster definitions.

Before Version 7.5, if the queue manager detected a problem in the local repository manager, it stopped the repository manager after a short interval. The queue manager kept running, processing application messages and requests to open queues, and publish or subscribe to topics.

With the repository manager stopped, the cache of cluster definitions available to the queue manager became more out of date. Over time, messages were routed to the wrong destination, and applications failed. Applications failed attempting to open cluster queues or publication topics that had not been propagated to the local queue manager.

Unless an administrator checked for repository messages in the error log, the administrator might not realize the cluster configuration had problems. If the failure was not recognized over an even longer time, and the queue manager did not renew its cluster membership, even more problems occurred. The instability affected all queue managers in the cluster, and the cluster appeared unstable.

From Version 7.5 onwards, IBM WebSphere MQ takes a different approach to cluster error handling. Rather than stop the repository manager and keep going without it, the repository manager reruns failed operations. If the queue manager detects a problem with the repository manager, it follows one of two courses of action.

1. If the error does not compromise the operation of the queue manager, the queue manager writes a message to the error log. It reruns the failed operation every 10 minutes until the operation succeeds. By default, you have five days to deal with the error; failing which, the queue manager writes a message to the error log, and shuts down. You can postpone the five day shutdown.
2. If the error compromises the operation of the queue manager, the queue manager writes a message to the error log, and shuts down immediately.

An error that compromises the operation of the queue manager is an error that the queue manager has not been able to diagnose, or an error that might have unforeseeable consequences. This type of error often results in the queue manager writing an FFST file. Errors that compromise the operation of the queue manager might be caused by a bug in IBM WebSphere MQ, or by an administrator, or a program, doing something unexpected, such as ending a IBM WebSphere MQ process.

The point of the change in error recovery behavior is to limit the time the queue manager continues to run with a growing number of inconsistent cluster definitions. As the number of inconsistencies in cluster definitions grows, the chance of abnormal application behavior grows with it.

The default choice of shutting down the queue manager after five days is a compromise between limiting the number of inconsistencies and keeping the queue manager available until the problems are detected and resolved.

You can extend the time before the queue manager shuts down indefinitely, while you fix the problem or wait for a planned queue manager shutdown. The five-day stay keeps the queue manager running through a long weekend, giving you time to react to any problems or prolong the time before restarting the queue manager.

Corrective actions

You have a choice of actions to deal with the problems of cluster error recovery. The first choice is to monitor and fix the problem, the second to monitor and postpone fixing the problem, and the final choice is to continue to manage cluster error recovery as in releases before Version 7.5.

1. Monitor the queue manager error log for the error messages AMQ9448 and AMQ5008, and fix the problem.

   AMQ9448 indicates that the repository manager has returned an error after running a command. This error marks the start of trying the command again every 10 minutes, and eventually stopping the queue manager after five days, unless you postpone the shutdown.

   AMQ5008 indicates that the queue manager was stopped because a IBM WebSphere MQ process is missing. AMQ5008 results from the repository manager stopping after five days. If the repository manager stops, the queue manager stops.
2. Monitor the queue manager error log for the error message AMQ9448, and postpone fixing the problem.

   If you disable getting messages from SYSTEM.CLUSTER.COMMAND.QUEUE, the repository manager stops trying to run commands, and continues indefinitely without processing any work. However, any handles that the repository manager holds to queues are released. Because the repository manager does not stop, the queue manager is not stopped after five days.

   Run an MQSC command to disable getting messages from SYSTEM.CLUSTER.COMMAND.QUEUE:

   ```
   ALTER QLOCAL(SYSTEM.CLUSTER.COMMAND.QUEUE) GET(DISABLED)
   ```

   To resume receiving messages from SYSTEM.CLUSTER.COMMAND.QUEUE run an MQSC command:

   ```
   ALTER QLOCAL(SYSTEM.CLUSTER.COMMAND.QUEUE) GET(ENABLED)
   ```
3. Revert the queue manager to the same cluster error recovery behavior as before Version 7.5.

   You can set a queue manager tuning parameter to keep the queue manager running if the repository manager stops.

   The tuning parameter is TolerateRepositoryFailure, in the TuningParameters stanza of the qm.ini file. To prevent the queue manager stopping, if the repository manager stops, set TolerateRepositoryFailure to TRUE; see Figure 90 on page 573.

   Restart the queue manager to enable the TolerateRepositoryFailure option.

   If a cluster error has occurred that prevents the repository manager starting successfully, and hence the queue manager from starting, set TolerateRepositoryFailure to TRUE to start the queue manager without the repository manager.

Special consideration

Before Version 7.5, some administrators managing queue managers that were not part of a cluster stopped the `amqrrmfa` process. Stopping `amqrrmfa` did not affect the queue manager.

Stopping `amqrrmfa` in Version 7.5 causes the queue manager to stop, because it is regarded as a queue manager failure. You must not stop the `amqrrmfa` process in Version 7.5, unless you set the queue manager tuning parameter, `TolerateRepositoryFailure`.

Example

```
TuningParameters:
    TolerateRepositoryFailure=TRUE
```

Figure 90. Set `TolerateRepositoryFailure` to `TRUE` in qm.ini

Related information:

Queue manager configuration files, qm.ini
A queue manager configuration file, qm.ini, contains information relevant to a specific queue manager.

AMQ9448

Change in behavior of MQS_REPORT_NOAUTH

In IBM WebSphere MQ Version 7.1, the default behavior of MQS_REPORT_NOAUTH was changed to *TRUE*.

This change causes UNIX platforms to behave like Windows, and log authorization failures to the error log.

Prior to IBM WebSphere MQ Version 7.1, this would only happen if you set MQS_REPORT_NOAUTH.

For more information, see MQS_REPORT_NOAUTH.

Command level changed to 750

The command level on platforms other than z/OS and IBM i changes to 750 in Version 7.5. z/OS and IBM i are at command level 710.

Related information:

MQCMDL_* (Command Levels)

CommandLevel (MQLONG)

Connect to multiple queue managers and use `MQCNO_FASTPATH_BINDING`

Applications that connect to queue managers using the `MQCNO_FASTPATH_BINDING` binding option might fail with an error and reason code `MQRC_FASTPATH_NOT_AVAILABLE`.

An application can connect to multiple queue managers from the same process. In releases earlier than Version 7.5, an application can set any one of the connections to `MQCNO_FASTPATH_BINDING`. In Version 7.5, only the first connection can be set to `MQCNO_FASTPATH_BINDING`. See "Fast path" on page 574 for the complete set of rules.

To assist with migration, you can set a new environment variable, `AMQ_SINGLE_INSTALLATION`. The variable reinstates the same behavior as in earlier releases, but prevents an application connecting to queue managers associated with other installations in the same process.

Fast path

On a server with multiple installations, applications using a fast path connection to IBM WebSphere MQ Version 7.1 or later must follow these rules:

1. The queue manager must be associated with the same installation as the one from which the application loaded the IBM WebSphere MQ run time libraries. The application must not use a fast path connection to a queue manager associated with a different installation. An attempt to make the connection results in an error, and reason code `MQRC_INSTALLATION_MISMATCH`.
2. Connecting non-fast path to a queue manager associated with the same installation as the one from which the application has loaded the IBM WebSphere MQ run time libraries prevents the application connecting fast path, unless either of these conditions are true:
   - The application makes its first connection to a queue manager associated with the same installation a fast path connection.
   - The environment variable, `AMQ_SINGLE_INSTALLATION` is set.
3. Connecting non-fast path to a queue manager associated with a Version 7.1 or later installation, has no effect on whether an application can connect fast path.
4. You cannot combine connecting to a queue manager associated with a Version 7.0.1 installation and connecting fast path to a queue manager associated with a Version 7.1, or later installation.

With `AMQ_SINGLE_INSTALLATION` set, you can make any connection to a queue manager a fast path connection. Otherwise almost the same restrictions apply:

- The installation must be the same one from which the IBM WebSphere MQ run time libraries were loaded.
- Every connection on the same process must be to the same installation. If you attempt to connect to a queue manager associated with a different installation, the connection fails with reason code `MQRC_INSTALLATION_MISMATCH`. Note that with `AMQ_SINGLE_INSTALLATION` set, this restriction applies to all connections, not only fast path connections.
- Only connect one queue manager with fast path connections.

Related information:

Binding options

2587 (0A1B) (RC2587): MQRC_HMSG_NOT_AVAILABLE

2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

Custom scripts

In IBM WebSphere MQ for Windows Version 7.5 custom scripts to install packages can fail as they have now been renamed.

Custom scripts to install IBM WebSphere MQ can be incomplete as they have been added or removed.

For a detailed explanation of each component, see Choosing what to install and select your platform.

Changes to data types

A number of data types have changed between IBM WebSphere MQ Version 7.0.1 to IBM WebSphere MQ Version 7.5 and new data types have been added. This topic lists the changes for data types that have a new current version in Version 7.5.

The current version of a data type is incremented if the length of a data type is extended by adding new fields. The addition of new constants to the values that can be set in a data type does not result in a change to the current version value.

Table 67 on page 575 lists the data types that have new versions. Click on the links to read about the new fields.

Table 67. New fields added to existing data types

Data type	New version	New fields
Channel definition	`MQCD_VERSION_10`	BatchDataLimit (MQLONG) DefReconnect (MQLONG) UseDLQ (MQLONG)
Channel exit	`MQCXP_VERSION_8`	MCAUserSource (MQLONG) pEntryPoints (PMQIEP)
Data conversion exit	`MQDXP_VERSION_2`	pEntryPoints(PMQIEP)
Pre-connect exit	`MQNXP_VERSION_2`	pEntryPoints(PMQIEP)
Publish exit publication context	`MQPBC_VERSION_2`	MsgDescPtr(PMQMD)
Publish exit	`MQPSXP_VERSION_2`	pEntryPoints (PMQIEP)
Cluster workload exit	`MQWXP_VERSION_4`	pEntryPoints (PMQIEP)

Default transmission queue restriction

The product documentation in previous versions of IBM WebSphere MQ warned about defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In Version 7.5, any attempt to set or use a default transmission queue that is defined as `SYSTEM.CLUSTER.TRANSMIT.QUEUE` results in an error.

In earlier versions of IBM WebSphere MQ no error was reported when defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. MQOPEN or MQPUT1 MQI calls that resulted in referencing the default transmission queue did not return an error. Applications might have continued working and failed later on. The reason for the failure was hard to diagnose.

The change ensures that any attempt to set the default transmission queue to `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, or use a default transmission queue set to `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, is immediately reported as an error.

Related reference:

"MQI and PCF reason code changes" on page 584
Reason codes that have changed and that affect some existing programs, are listed.

Related information:

AMQ8520: The queue name supplied is not valid for DEFXMITQ.

Display channel and cluster status: Switching

From IBM WebSphere MQ Version 7.5, a cluster-sender channel that is switching its configuration to a different cluster transmission queue has a new channel state: Switching.

When migrating from a release before Version 7.5 to Version 7.5 or later, existing application programs are not affected by the new state.

System management programs that monitor channel or cluster status might receive the new state as a result of a inquiry.

The state is set during the short interval while the channel modifies the destination transmission queue that messages are stored on. Before the switching state is set, messages are stored on the previously associated transmission queue. After the switching state, messages are stored on the newly configured transmission queue. The channel enters the switching state if a cluster-sender channel is starting, a configuration change is required, and the conditions for starting the switch are met.

Related information:

Channel states
A channel can be in one of many states at any time. Some states also have substates. From a given state a channel can move into other states.

Cluster transmission queues and cluster-sender channels
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

DISPLAY CHSTATUS

Inquire Channel Status (Response)

Inquire Cluster Queue Manager (Response)

MQCHS_* (Command format Channel Status)

dspmqver

New types of information are displayed by **dspmqver** to support multiple installations. The changes might affect existing administrative scripts you have written to manage IBM WebSphere MQ.

The changes in output from **dspmqver** that might affect existing command scripts that you have written are twofold:

1. Version 7.5 has extra `-f` field options. If you do not specify a `-f` option, output from all the options is displayed. To restrict the output to the same information that was displayed in earlier releases, set the `-f` option to a value that was present in the earlier release. Compare the output for `dspmqver` in Figure 91 and Figure 92 with the output for `dspmqver -f 15` in Figure 93 on page 577.

```
dspmqver
Name:        WebSphere MQ
Version:     7.0.1.6
CMVC level:  p701-L110705
BuildType:   IKAP - (Production)
```

Figure 91. Default **dspmqver** *options in IBM WebSphere MQ Version 7.0.1*

```
dspmqver
Name:        WebSphere MQ
Version:     7.1.0.0
Level:       p000-L110624
BuildType:   IKAP - (Production)
Platform:    WebSphere MQ for Windows
Mode:        32-bit
O/S:         Windows XP, Build 2600: SP3
InstName:    110705
InstDesc:    July 5 2011
InstPath:    C:\Program Files\IBM\WebSphere MQ_110705
DataPath:    C:\Program Files\IBM\WebSphere MQ
Primary:     No
MaxCmdLevel: 710

Note there are a number (1) of other installations,
use the '-i' parameter to display them.
```

Figure 92. Default **dspmqver** *options in IBM WebSphere MQ Version 7.5*

```
dspmqver -f 15
Name:        WebSphere MQ
Version:     7.1.0.0
Level:       p000-L110624
BuildType:   IKAP - (Production)
```

Figure 93. **dspmqver** *with option to make IBM WebSphere MQ Version 7.5 similar to IBM WebSphere MQ Version 7.0.1*

2. The heading of the build level row has changed from `CMVC level:` to `Level:`.

Related information:

dspmqver

Exits and installable services

When migrating to IBM WebSphere MQ Version 7.5 for a distributed platform, if you install IBM WebSphere MQ in a non-default location, you must update your exits and installable services. Data conversion exits generated using the **crtmqcvx** command must be regenerated using the updated command.

When writing new exits and installable services, you do not need to link to any of the following IBM WebSphere MQ libraries:

- mqmzf
- mqm
- mqmvx
- mqmvxd
- mqic
- mqutl

For more information about updating existing exits, and writing exits and installable services that do not link to the libraries, see Writing and compiling exits and installable services.

Fewer IBM WebSphere MQ MQI client log messages

A IBM WebSphere MQ MQI client used to report every failed attempt to connect to a queue manager when processing a connection name list. From Version 7.5, only if the failure occurs with the last connection in the list is a message written to the queue manager error log.

Reporting the last failure and no others, reduces growth of the queue manager error log.

GSKit: Changes from GSKit V7.0 to GSKit V8.0

For distributed platforms, GSKit V8.0 is integrated with IBM WebSphere MQ. In versions of IBM WebSphere MQ prior to Version 7.1, you installed GSKit separately. GSKit V8.0 was included as an alternative to GSKit V7.0 in IBM WebSphere MQ Version 7.0.1; it is now the only version of GSKit provided with IBM WebSphere MQ. Some functions in GSKit V8.0 are different to the functions in GSKit V7.0.

Review the following list of changes.

GSKit: Some FIPS 140-2 compliant channels do not start:

Three CipherSpecs are no longer FIPS 140-2 compliant. If a client or queue manager is configured to require FIPS 140-2 compliance, channels that use the following CipherSpecs do not start after migration.

- FIPS_WITH_DES_CBC_SHA
- FIPS_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_DES_CBC_SHA

To restart a channel, alter the channel definition to use a FIPS 140-2 compliant CipherSpec. Alternatively, configure the queue manager, or the client in the case of a IBM WebSphere MQ MQI client, not to enforce FIPS 140-2 compliance.

Earlier versions of IBM WebSphere MQ enforced an older version of the FIPS 140-2 standard. The following CipherSpecs were considered FIPS 140-2 compliant in earlier versions of IBM WebSphere MQ and are also compliant in this version:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (deprecated)
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA256 (only when AltGSKit version 8 is used with Fix Pack 7.0.1.4 or later)
- TLS_RSA_WITH_AES_256_CBC_SHA256 (only when AltGSKit version 8 is used with Fix Pack 7.0.1.4 or later)

Use these CipherSpecs if you want IBM WebSphere MQ to interoperate in a FIPS 140-2 compliant manner with earlier versions.

Previous IBM WebSphere MQ releases enforced an older version of the FIPS 140-2 standard. The following CipherSpecs were considered FIPS 140-2 compliant by previous IBM WebSphere MQ releases and are also considered compliant by IBM WebSphere MQ:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (deprecated)
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA

Use these CipherSpecs if you need IBM WebSphere MQ version 7.1 to interoperate in a FIPS 140-2 compliant manner with earlier IBM WebSphere MQ releases.

Related information:

AMQ9196

Federal Information Processing Standards (FIPS)

Federal Information Processing Standards

FipsRequired (MQLONG)

MQSSLFIPS
MQSSLFIPS specifies whether only FIPS-certified algorithms are to be used if cryptography is carried out in WebSphere MQ. The values are the same as for the SSLFIPS parameter of the ALTER QMGR command.

Specifying that only FIPS-certified CipherSpecs are used at run time on the MQI client

SSLFIPSRequired (MQLONG)

GSKit: Certificate Common Name (CN) not mandatory:

In GSKit V8.0, the **iKeyman** command accepts any element of the distinguished name (DN), or a form of the subject alternative name (SAN). It does not mandate that you provide it with a common name. In GSKit V7.0, if you create a self-signed certificate using the **iKeyman** command you had to specify a common name.

The implication is that applications searching for a certificate might not able to assume that a certificate has a common name. You might need to review how applications search for certificates, and how applications handle errors involving the common name. Alternatively, you might choose to check that all self-signed certificates are given common names.

Some other certificate tools that you might also be using, do not require a common name. It is therefore likely the change to GSKit is not going to cause you a problem.

Related information:

Distinguished Names

GSKit: Commands renamed:

The command name **gsk7cmd** is replaced with **runmqckm**; **gsk7ikm** is replaced with **strmqikm**, and **gsk7capicmd** is replaced with **runmqakm** . All the commands start the GSKit V8.0 certificate administration tools, and not the GSKit V7.0 tools.

IBM WebSphere MQ Version 7.5 does not use a machine-wide shared installation of GSKit: instead it uses a private GSKit installation in the IBM WebSphere MQ installation directory. Each IBM WebSphere MQ Version 7.5 installation can use a different GSKit version. To display the version number of GSKit embedded in a particular WebSphere MQ installation, run the **dspmqver** command from that installation as shown in the following table:

Table 68. Renamed GSKit commands

Platform	GSKit V7.0 command	GSKit V8.0 command
UNIX and Linux	**gsk7cmd**	**runmqckm**
UNIX and Linux	**gsk7ikm**	**strmqikm**
Windows,UNIX and Linux	**gsk7capicmd**	**runmqakm**
Windows,UNIX and Linux	**gsk7ver**	**dspmqver -p 64 -v**

Note: Do not use the **gsk8ver** command to display the GSKit version number: only the **dspmqver** command will show the correct GSKit version number for IBM WebSphere MQ Version 7.5.

Related information:

runmqckm, and runmqakm commands

Using iKeyman, iKeycmd, runmqakm, and runmqckm

dspmqver

GSKit: The iKeyman commands to insert a certificate do not check that all required CA certificates are present:

The **iKeyman** command in GSKit V8.0 does not validate a certificate when it is inserted into a key repository. **iKeyman** in GSKit V7.0, validated a certificate before it inserted the certificate into a certificate store.

The implication is, that if you create a certificate using the **iKeyman** in GSKit V8.0, all the necessary intermediate and root CA certificates might not be present, or they might have expired; when the certificate is checked it might fail.

Necessary certificates might be missing, or have expired. This can cause SSL and TLS connections to fail with error AMQ9633.

Related information:

Certificate validation and trust policy design on UNIX, Linux and Windows systems

GSKit: PKCS#11 and JRE addressing mode:

If you use **iKeyman** or **iKeycmd** to administer certificates and keys for PKCS#11 cryptographic hardware note that the addressing mode of the JRE for these tools has changed.

On the following platforms the JRE was 32 bit however in IBM WebSphere MQ Version 7.5 it is now 64 bit only. Where it has changed you might need to install additional PKCS#11 drivers appropriate for the addressing mode of the **iKeyman** and **iKeycmd** JRE. This is because the PKCS#11 driver must use the same addressing mode as the JRE. The following table shows the IBM WebSphere MQ Version 7.5 JRE addressing modes.

Table 69. IBM WebSphere MQ Version 7.5 JRE addressing modes

Platform	JRE Addressing Mode
Windows (32 bit or 64 bit)	32
Linux for System x 32 bit	32
Linux for System x 64 bit	64
Linux for System p	64
Linux for System z	64
HP-UX	64
Solaris Sparc	64
Solaris x86-64	64
AIX	64

GSKit: Import of a duplicate PKCS#12 certificate:

In GSKit V8.0, the **iKeyman** command does not report an attempt to import a duplicate PKCS#12 certificate as an error. In GSKit V7.0, the **iKeyman** command reported an error. In neither version is a duplicate certificate imported.

For GSKIT V8.0, a duplicate certificate is a certificate with the same label and public key.

The implication is that if some of the issuer information is different, but the name and public key are the same, the changes are not imported. The correct way to update a certificate is to use the `-cert -receive` option, which replaces an existing certificate.

gskcapicmd does not allow or ignore duplicates on import in this way.

Related information:
Importing a personal certificate into a key repository on UNIX, Linux or Windows systems

GSKit: Certificate stores created by `iKeyman` and `iKeycmd` no longer contain CA certificates:

The **`iKeyman`** and **`iKeycmd`** utilities in GSKit V8.0 create a certificate store without adding pre-defined CA certificates to the store. To create a working certificate store, you must now add all the certificates that you require and trust. In GSKit V7.0 **`iKeyman`** and **`iKeycmd`** created a certificate store that already contained CA certificates.

Existing data bases created by GSKit V7.0 are unaffected by this change.
Related information:
Adding default CA certificates into an empty key repository, on UNIX, Linux or Windows systems with GSKit version 8.0

GSKit: Password expiry to key database deprecated:

In GSKit V8.0, the password expiry function in **`iKeyman`** continues to work the same as in GSKit V7.0, but it might be withdrawn in future versions of GSKit.

Use the file system protection provided with the operating system to protect the key database and password stash file.

Linux: Recompile C++ applications and update run time libraries:

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.
Related tasks:
"Linux: Rebuilding a C++ application" on page 491
C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

GSKit: Signature algorithm moved out of settings file:

In GSKit V8.0, the default signature algorithm used when creating self-signed certificates or certificate requests or selected in the creation dialogs is passed as a command-line parameter. In GSKit V7.0, the default signature algorithm was specified in the settings file.

The change has very little effect: it causes a different default signature algorithm to be selected. It does not alter the selection of a signature algorithm.

Related information:

runmqckm and runmqakm options

Creating a self-signed personal certificate on UNIX, Linux, and Windows systems

GSKit: Signed certificate validity period not within signer validity:

In GSKit V8.0, the **iKeyman** command does not check whether the validity period of a resulting certificate is within the validity period of the signed certificate. In GSKit V7.0, **iKeyman** checked that the validity period of the resulting certificate was within the validity period of the signed certificate.

The IETF RFC standards for SSL/TLS allow a certificate whose validity dates extend beyond those of its signer. This change to GSKit brings it into line with those standards. The check is whether the certificate is issued within the validity period of the signer, and not whether it expires within the validity period of the signer.

Related information:

How SSL and TLS provide identification, authentication, confidentiality, and integrity

GSKit: Stricter default file permissions:

The default file permissions set by **runmqckm** and **strmqikm** in IBM WebSphere MQ Version 7.5 on UNIX and Linux are stricter than the permissions that are set by **runmqckm**, **strmqikm**, **gsk7cmd**, and **gsk7ikm** in earlier releases of IBM WebSphere MQ.

The permissions set by **runmqckm** and **strmqikm** in IBM WebSphere MQ Version 7.5 permit only the creator to access the UNIX and Linux SSL/TLS key databases. The **runmqckm**, **strmqikm**, **gsk7cmd**, and **gsk7ikm** tools in earlier releases of IBM WebSphere MQ set world-readable permissions, making the files liable to theft and impersonation attacks.

The permissions set by **gsk7capicmd**, in earlier releases of IBM WebSphere MQ, and **runmqakm** in IBM WebSphere MQ Version 7.5, permit only the creator to access UNIX and Linux SSL/TLS key databases.

The migration of SSL/TLS key databases to Version 7.5 does not alter their access permissions. In many cases, administrators set more restrictive access permissions on these files to overcome the liability to theft and impersonation attacks; these permissions are retained.

The default file permissions set on Windows are unchanged. Continue to tighten up the access permissions on SSL/TLS key database files on Windows after creating the files with **runmqckm** or **strmqikm**.

Related information:

Accessing and securing your key database files on Windows

Accessing and securing your key database files on UNIX and Linux systems

Java: Change in behavior of default value of MQEnvironment.userID

Change caused when using CLIENT transport for a channel not have a security exit defined.

If a IBM WebSphere MQ classes for Java application is connecting to a queue manager, using the CLIENT transport through a channel that does not have a security exit defined, and the MQEnvironment.userID field is left at its default value of the empty string (""), the IBM WebSphere MQ classes for Java application queries the value of the Java System Property `user.name` and passes this to the queue manager for authorization as part of the MQQueueManager constructor.

If the user specified by the Java System Property `user.name` is not authorized to access the queue manager, the MQQueueManager constructor throws an MQException containing Reason Code MQRC_NOT_AUTHORIZED.

JMS: Change in behaviour of the default user identifier value

Change caused when using CLIENT transport for a channel that does not have a security exit defined.

If a IBM WebSphere MQ classes for JMS application is connecting to a queue manager, using the CLIENT transport through a channel that does not have a security exit defined, and no user identifier is specified, by calling `ConnectionFactory.createConnection()`, the IBM WebSphere MQ classes for JMS application queries the value of the Java System Property `user.name` and passes this to the queue manager for authorization as part of the call to create a connection from a connection factory object. This behaviour also occurs when calling `ConnectionFactory.createConnection(String, String)` and passing a blank or null value for the first parameter **`userID`**.

If the user specified by the Java System Property `user.name` is not authorized to access the queue manager, a JMSException containing Reason Code MQRC_NOT_AUTHORIZED is thrown.

Java: Different message property data type returned

If the data type of a message property is set, the same data type is returned when the message is received. In some circumstances in IBM WebSphere MQ Version 7.0.1, properties set with a specific type were returned with the default type String.

The change affects Java applications that used the MQRFH2 class, and retrieved properties using the `getFieldValue` method.

You can write a message property in Java using a method such as `setIntFieldValue`. In IBM WebSphere MQ Version 7.0.1 the property is written into an `MQRFH2` header with a default type of String. When you retrieve the property with the `getFieldValue` method, a String object is returned.

The change is that now the correct type of object is returned, in this example the type of object returned is Integer.

If your application retrieves the property with the `getIntFieldValue` method, there is no change in behavior; an int is returned. If property is written to the `MQRFH2` header by some other means, and the data type is set, then `getFieldValue` returns the correct type of object.

Related information:

Class MQRFH2

JMS: Reason code changes

Some reason codes returned in JMS exceptions have changed. The changes affect `MQRC_Q_MGR_NOT_AVAILABLE` and `MQRC_SSL_INITIALIZATION_ERROR`.

In earlier releases of IBM WebSphere MQ, if a JMS application call fails to connect, it receives an exception with a reason code 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE. In Version 7.5, it can still receive `MQRC_Q_MGR_NOT_AVAILABLE`, or one of the following more specific reason codes.

- 2537 (09E9) (RC2537): MQRC_CHANNEL_NOT_AVAILABLE
- 2538 (09EA) (RC2538): MQRC_HOST_NOT_AVAILABLE
- 2539 (09EB) (RC2539): MQRC_CHANNEL_CONFIG_ERROR
- 2540 (09EC) (RC2540): MQRC_UNKNOWN_CHANNEL_NAME

Similarly, when trying to connect, a JMS application might have received 2393 (0959) (RC2393): MQRC_SSL_INITIALIZATION_ERROR. In Version 7.5, it can still receive `MQRC_SSL_INITIALIZATION_ERROR`, or a more specific reason code, such as 2400 (0960) (RC2400): MQRC_UNSUPPORTED_CIPHER_SUITE, that identifies the cause of the SSL initialization error.

JMS: ResourceAdapter object configuration

When a WebSphere Application Server connects to IBM WebSphere MQ it creates Message Driven Beans (MDBs) using JMS connections. These MDBs can no longer share one JMS connection. The configuration of ResourceAdapter object is migrated so that there is a single MDB for each JMS connection.

Changed ResourceAdapter properties

`connectionConcurrency`
: The maximum number of MDBs to share a JMS connection. Sharing connections is not possible and this property always has the value 1. Its previous default value was 5.

`maxConnections`
: This property is the number of JMS connections that the resource adapter can manage. In Version 7.5, it also determines the number of MDBs that can connect because each MDB requires one JMS connection. The default value of `maxConnections` is now `50`. Its previous default value was `10`.

If `connectionConcurrency` is set to a value greater than 1, the maximum number of connections supported by the resource adapter is scaled by the value of `connectionConcurrency`. For example, if `maxConnections` is set to `2` and `connectionConcurrency` is set to `4`, the maximum number of connections supported by the resource adapter is 8. As a result, `connectionConcurrency` is set to `1` and `maxConnections` is set to `8`.

If `connectionConcurrency` is set to a value greater than `1`, it is adjusted automatically. To avoid automatic adjustment, set `connectionConcurrency` to `1`. You can then set `maxConnections` to the value you want.

The scaling mechanism ensures that sufficient connections are available for existing deployments whether you have changed them in your deployment, configuration, or programs.

If the adjusted `maxConnections` value exceeds the `MAXINST` or `MAXINSTC` attributes of any used channel, previously working deployments might fail.

The default value of both channel attributes equates to unlimited. If you changed them from the default value, you must ensure that the new `maxConnections` value does not exceed `MAXINST` or `MAXINSTC`.

Related information:

Configuration of the ResourceAdapter object

Maximum instances (MAXINST)

Maximum instances per client (MAXINSTC)

MQI and PCF reason code changes

Reason codes that have changed and that affect some existing programs, are listed.

`MQRC_NOT_OPEN_FOR_INPUT`
: In IBM WebSphere MQ Version 7.0 a queue opened with `MQOO_OUTPUT`, and then browsed, returned an error with the wrong reason-code, `MQRC_NOT_OPEN_FOR_INPUT`. The correct reason-code, `MQRC_NOT_OPEN_FOR_BROWSE`, was issued by Version 6.0 and earlier. Version 7.5 correctly returns an error with the same reason code as Version 6.0, `MQRC_NOT_OPEN_FOR_BROWSE`.

`MQRC_DEF_XMIT_Q_USAGE_ERROR`
: The product documentation in previous versions of IBM WebSphere MQ warned about defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In Version 7.5, an attempt to open the default transmission queue, defined as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, results in the error `MQRC_DEF_XMIT_Q_USAGE_ERROR`.

`MQRC_FASTPATH_NOT_AVAILABLE`
: An application that connects to multiple queue managers in the same process and uses `MQCNO_FASTPATH_BINDING` might fail with an error and reason code `MQRC_FASTPATH_NOT_AVAILABLE`; see “Connect to multiple queue managers and use `MQCNO_FASTPATH_BINDING`” on page 573.

MQRCCF_DEF_XMIT_Q_CLUS_ERROR

The product documentation in previous versions of IBM WebSphere MQ warned about defining the default transmission queue as SYSTEM.CLUSTER.TRANSMIT.QUEUE. In Version 7.5, an attempt to alter the queue manager attribute **DEFXMITQ** to SYSTEM.CLUSTER.TRANSMIT.QUEUE results in an error. The PCF reason code is 3269 (0CC5) (RC3269): MQRCCF_DEF_XMIT_Q_CLUS_ERROR.

Related reference:

"Connect to multiple queue managers and use MQCNO_FASTPATH_BINDING" on page 573
Applications that connect to queue managers using the MQCNO_FASTPATH_BINDING binding option might fail with an error and reason code MQRC_FASTPATH_NOT_AVAILABLE.

Related information:

2036 (07F4) (RC2036): MQRC_NOT_OPEN_FOR_BROWSE

2037 (07F5) (RC2037): MQRC_NOT_OPEN_FOR_INPUT

2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

3269 (0CC5) (RC3269): MQRCCF_DEF_XMIT_Q_CLUS_ERROR

AMQ8520

Publish/Subscribe: Delete temporary dynamic queue

If a subscription is associated with a temporary dynamic queue, when the queue is deleted, the subscription is deleted. This changes the behavior of incorrectly written publish/subscribe applications migrated from Version 6.0. Publish/subscribe applications migrated from WebSphere Message Broker are unchanged. The change does not affect the behavior of integrated publish/subscribe applications, which are written using the MQI publish/subscribe interface.

- In IBM WebSphere MQ Version 6.0 the product documentation stated "The subscriber queue must not be a temporary dynamic queue"; see Queue name. But the injunction was not enforced, although the configuration is not supported.
- In WebSphere Message Broker and WebSphere Event Broker version 6.0, and WebSphere Message Broker version 6.1, you could create a subscription that used a temporary dynamic queue as the subscriber queue. If the queue was deleted, the subscription was deleted with it.
- In IBM WebSphere MQ Version 7.0, if you migrate or create a queued publish/subscribe application that uses MQRFH1, it behaves the same as IBM WebSphere MQ Version 6.0. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is not deleted, as in IBM WebSphere MQ Version 6.0. The lack of a subscriber queue results in any matching publications ending up on the dead letter queue. This behavior is the same as IBM WebSphere MQ Version 6.0. MQRFH1 publish/subscribe applications are typically migrated from IBM WebSphere MQ Version 6.0.
- In IBM WebSphere MQ Version 7.0.1 from fix pack 7.0.1.6 onwards, and in Version 7.5, in the same MQRFH1 queued publish/subscribe case, if the temporary dynamic queue is deleted, the subscription is deleted. This change prevents a buildup of publications from a subscription without a subscriber queue ending up on the dead letter queue. It is a change from IBM WebSphere MQ Version 6.0.
- In Version 7.5, if you migrate or create a queued publish/subscribe application that uses MQRFH2, it behaves the same as WebSphere Event Broker and WebSphere Message Broker. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is deleted, as in WebSphere Event Broker and WebSphere Message Broker. MQRFH2 publish/subscribe applications are typically migrated from WebSphere Event Broker or WebSphere Message Broker.
- In IBM WebSphere MQ Version 7.5, if you create a durable subscription using integrated publish/subscribe, you cannot define a temporary dynamic queue as the destination for its matching publications.
- In IBM WebSphere MQ Version 7.5, you can create a managed, non-durable subscription using integrated publish/subscribe, which creates a temporary dynamic queue as the destination for matching publications. The subscription is deleted with the queue.

Summary

In Version 7.5, you cannot create a temporary dynamic queue as the destination for publications for a durable subscription using the integrated publish/subscribe interface.

In the current fix level of Version 7.5, if you use either of the queued publish/subscribe interfaces, MQRFH1 or MQRFH2, the behavior is the same. You can create a temporary dynamic queue as the subscriber queue, and if the queue is deleted, the subscription is deleted with it. Deleting the subscription with the queue retains the same the supported behavior of IBM WebSphere MQ Version 6.0, WebSphere Event Broker, and WebSphere Message Broker applications. It modifies the unsupported behavior of IBM WebSphere MQ Version 6.0 applications.

Queue manager logs: Default sizes increased

The default size of a queue manager log files has been changed to 4096. The AMQERR*nn*.log has increased from 256 KB to 2 MB on UNIX, Linux, and Windows platforms. The change affects both new and migrated queue managers.

Queue manager log

In IBM WebSphere MQ Version 7.5 default log size is 4096. For more information on setting non-default values, see The IBM WebSphere MQ configuration file, mqs.ini.

Queue manager error log

Override the change by setting the environment variable MQMAXERRORLOGSIZE, or setting ErrorLogSize in the QMErrorLog stanza in the qm.ini file.

The change increases the number of error messages that are saved in the error logs.

Related information:

Queue manager error logs
Use the Extended queue manager properties page from the WebSphere MQ Explorer, or the QMErrorLog stanza in the qm.ini file to tailor the operation and contents of queue manager error logs.

SSLPEER and SSLCERTI changes

IBM WebSphere MQ Version 7.5 obtains the Distinguished Encoding Rules (DER) encoding of the certificate and uses it to determine the subject and issuer distinguished names. The subject and issuer distinguished names are used in the SSLPEER and SSLCERTI fields. A SERIALNUMBER attribute is also included in the subject distinguished name and contains the serial number for the certificate of the remote partner. Some attributes of subject and issuer distinguished names are returned in a different sequence from previous releases.

The change to subject and issuer distinguished names affects channel security exits. It also affects aplications which depend upon the subject and issuer distinguished names that are returned by the PCF programming interface. Channel security exits and applications that set or query SSLPEER and SSLCERTI must be examined, and possibly changed. The fields that are affected are listed in Table 70 on page 587 and Table 71 on page 587.

Table 70. Channel status fields affected by changes to subject and issuer distinguished names

Channel status attribute	PCF channel parameter type
SSL Peer (SSLPEER)	MQCACH_SSL_SHORT_PEER_NAME
SSLCERTI	MQCACH_SSL_CERT_ISSUER_NAME

Table 71. Channel data structures affected by changes to subject and issuer distinguished names

Channel data structure	Field
MQCD - Channel definition	SSLPeerNamePtr (MQPTR)
MQCXP - Channel exit parameter	SSLRemCertIssNamePtr (PMQVOID)

Existing peer name filters specified in the SSLPEER field of a channel definition are not affected. They continue to operate in the same manner as in earlier releases. The peer name matching algorithm has been updated to process existing SSLPEER filters. It is not necessary to alter any channel definitions.

Related information:

Channel security exit programs

Telemetry: Installer integrated with IBM WebSphere MQ

From Version 7.1, IBM WebSphere MQ Telemetry is no longer installed separately from IBM WebSphere MQ. It is installed as a component of IBM WebSphere MQ. If you installed IBM WebSphere MQ Telemetry with Version 7.0.1, you must uninstall it before installing the current version of the product.

You can install IBM WebSphere MQ Telemetry at the same time as IBM WebSphere MQ, or you can rerun the installer and install IBM WebSphere MQ Telemetry at a later time.

Related information:

Installing WebSphere MQ Telemetry
From IBM WebSphere MQ Version 7.1, IBM WebSphere MQ Telemetry is a component of the main IBM WebSphere MQ product, and is no longer a separate plug-in. There are two main methods of installing IBM WebSphere MQ Telemetry: selecting IBM WebSphere MQ Telemetry in a custom installation when installing IBM WebSphere MQ, or modifying an existing IBM WebSphere MQ installation and selecting IBM WebSphere MQ Telemetry.

Uninstalling WebSphere MQ Telemetry version 7.0.1 components

IBM WebSphere MQ Explorer changes

IBM WebSphere Eclipse Platform is no longer shipped with IBM WebSphere MQ; it is not required to run MQ Explorer. The change makes no difference to administrators who run MQ Explorer. For developers who run MQ Explorer in an Eclipse development environment, a change is necessary. You must install and configure a separate Eclipse environment to be able to switch between MQ Explorer and other perspectives.

Packaging changes

In versions of IBM WebSphere MQ earlier than Version 7.5, you can select the Workbench mode preference in MQ Explorer. In workbench mode, you could switch to the other perspectives installed in the WebSphere Eclipse Platform. You can no longer set the Workbench mode preference, because the WebSphere Eclipse Platform is not shipped with MQ Explorer in Version 7.5.

To switch between MQ Explorer and other perspectives, you must install MQ Explorer into your own Eclipse environment or into an Eclipse-based product. You can then switch between perspectives. For example, you can develop applications using IBM WebSphere MQ classes for JMS or IBM WebSphere MQ Telemetry applications; see Creating your first MQ Telemetry Transport publisher application using Java

If you installed extensions to previous versions of MQ Explorer, such as SupportPacs or WebSphere Message Broker Explorer, you must reinstall compatible versions of the extensions after upgrading MQ Explorer to Version 7.5.

If you continue to run IBM WebSphere MQ Version 7.0.1 on the same server as IBM WebSphere MQ Version 7.5, and you use MQ Explorer, each installation uses its own installation of MQ Explorer. When you uninstall Version 7.0.1, its version of MQ Explorer is uninstalled. To remove IBM WebSphere Eclipse Platform, uninstall it separately. The workspace is not deleted.

Test result migration

Test results are not migrated from version to version. To view any test results, you must rerun the tests.

Related information:

Multiple installations of WebSphere MQ Explorer

AIX: Shared objects

In IBM WebSphere MQ for AIX Version 7.5 the `.a` shared objects in the `lib64` directory contains both the 32 bit and 64 bit objects. A symlink to the `.a` file is also placed in the `lib` directory. The AIX loader can then correctly pick up the correct object for the type of application being run.

This means that IBM WebSphere MQ applications can run with the `LIBPATH` containing either the `lib` or `lib64` directory, or both.

AIX: `/usr/lpp/mqm` symbolic link removed

Before Version 6.0, IBM WebSphere MQ placed a symbolic link in `/usr/lpp/mqm` on AIX. The link ensured queue managers and applications migrated from IBM WebSphere MQ versions before Version 5.3 continued to work, without change. The link is not created in Version 7.5.

In version 5.0, IBM WebSphere MQ for AIX was installed into `/usr/lpp/mqm`. That changed in Version 5.3 to `/usr/mqm`. A symbolic link was placed in `/usr/lpp/mqm`, linking to `/usr/mqm`. Existing programs and scripts that relied on the installation into `/usr/lpp/mqm` continued to work unchanged. That symbolic link has been removed in Version 7.5, because you can now install IBM WebSphere MQ in any directory. Applications and command scripts are affected by the change.

The effect on applications is no different to the effect of migrating on other UNIX and Linux platforms. If the installation is made primary, then symbolic links to the IBM WebSphere MQ link libraries are placed in `/usr/lib`. Most applications migrated from earlier IBM WebSphere MQ versions search the default search path, which normally includes `/usr/lib`. The applications find the symbolic link to the IBM WebSphere MQ load libraries in `/usr/lib`.

If the installation is not primary, then you must configure the correct search path to load the IBM WebSphere MQ link libraries. If you choose to run **`setmqenv`**, IBM WebSphere MQ places the IBM WebSphere MQ link library path into `LIBPATH`. Unless the application is configured not to search the `LIBPATH`, if for example it is a `setuid` or `setgid` application, then the IBM WebSphere MQ library is loaded successfully; see “UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5” on page 488.

If you have written command scripts that run IBM WebSphere MQ commands, you might have coded explicit paths to the directory tree where IBM WebSphere MQ was installed. You must modify these command scripts. You can run **`setmqenv`** to create the correct environment to run the command scripts. If you have set the installation as primary, you do not have to specify the path to the command.

Related tasks:

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Related information:

"Loading IBM WebSphere MQ libraries" on page 350

AIX, HP-UX, and Solaris: Building applications for TXSeries

You must rebuild IBM WebSphere MQ applications that link to TXSeries.

Before Version 7.5, IBM WebSphere MQ applications that used the TXSeries CICS support loaded the IBM WebSphere MQ library, `mqz_r`. From Version 7.5 those applications must load the IBM WebSphere MQ library, `mqzi_r` instead. You must change your build scripts accordingly.

Version 7.5 of `mqz_r` includes code to load a different version of the IBM WebSphere MQ library. IBM WebSphere MQ loads a different version of the IBM WebSphere MQ library, if it detects that the queue manager the application is connected to is associated with a different installation to the one from which the library was loaded. `mqzi_r` does not include the additional code. When using TXSeries, the application must run with the IBM WebSphere MQ library it loaded, and not a different library loaded by IBM WebSphere MQ. For this reason, IBM WebSphere MQ applications that use the IBM WebSphere MQ TXSeries support must load the `mqzi_r` library, and not the `mqz_r` library.

An implication of applications loading `mqzi_r` is that the application must load the correct version of `mqzi_r`. It must load the one from the installation that is associated with the queue manager that the application is connected to.

Related information:

Building libraries for use with TXSeries for Multiplatforms version 5

AIX: TXSeries CICS support

HP-UX: TXSeries CICS support

Solaris: TXSeries CICS support

Linux: Recompile C++ applications and update run time libraries

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.

Related tasks:

"Linux: Rebuilding a C++ application" on page 491
C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

Linux: Increased shared memory allocation required

The maximum amount of shared memory (`SHMMAX`) to allocate on Linux systems was omitted from the Version 7.0 product documentation. The default system allocation is 32 MB. IBM WebSphere MQ starts by allocating 64 MB and increases its allocation on demand by doubling its previous allocation. On a production system set `SHMMAX` to at least 256 MB to accommodate additional allocations.

Related information:

Additional settings for installing WebSphere MQ on Linux systems
Use this topic to when configuring IBM WebSphere MQ on Linux systems.

UNIX and Linux: `crtmqlnk` and `dltmqlnk` removed

The **`crtmqlnk`** and **`dltmqlnk`** commands are not present in this version of the product. In versions prior to Version 7.1, the commands created symbolic links in subdirectories of `/usr`. From Version 7.1 onwards, you must use the **`setmqinst`** command instead.

Related reference:

"UNIX and Linux: `/usr` symbolic links removed" on page 591
On all UNIX and Linux platforms, the links from the `/usr` file system are no longer made automatically. In order to take advantage of these links, you must set an installation as the primary installation or set the links up manually.

Related information:

Changing the primary installation
You can use the **`setmqinst`** command to set or unset an installation as the primary installation.

setmqinst

UNIX and Linux: Message catalogs moved

IBM WebSphere MQ message catalogs are no longer stored in the system directories in Version 7.5. To support multiple installations, copies of the message catalogs are stored with each installation. If you want messages only in the locale of your system, the change has no affect on your system. If you have customized the way the search procedure selects a message catalog, then the customization might no longer work correctly.

Set the `LANG` environment variable to load a message catalog for a different language from the system locale.

Related information:

Displaying messages in your national language on UNIX and Linux systems
To display messages from a different national language message catalog, you must install the appropriate catalog and set the **`LANG`** environment variable.

UNIX and Linux: MQ services and triggered applications

IBM WebSphere MQ Version 7.5 has been configured to allow both `LD_LIBRARY_PATH` and `$ORIGIN` to work for MQ services and triggered applications. For this reason MQ Services and triggered applications have been changed so that they run under the user ID who started the queue manager and not `setuid` or `setgid`.

If any files used by the service were previously restricted to certain users, then they might not be accessible by the user ID who started the queue manager. Resources used by MQ services or triggered applications must be adjusted as appropriate.

Note:

- On AIX, `LD_LIBRARY_PATH` is also known as `LIBPATH` and `$ORIGIN` is not supported.
- On HP-UX, `LD_LIBRARY_PATH` is also known as `SHLIB_PATH`.

Related information:

Working with services

Starting WebSphere MQ applications using triggers

UNIX and Linux: `ps -ef | grep amq` interpretation

The interpretation of the list of IBM WebSphere MQ processes that results from filtering a scan of UNIXor Linux processes has changed. The results can show IBM WebSphere MQ processes running for multiple installations on a server. Before Version 7.5, the search identified IBM WebSphere MQ processes running on only a single installation of IBM WebSphere MQ on a UNIXor Linux server.

The implications of this change depend on how the results are qualified and interpreted, and how the list of processes is used. The change affects you, only if you start to run multiple installations on a single server. If you have incorporated the list of IBM WebSphere MQ processes into administrative scripts or manual procedures, you must review the usage.

Examples

The following two examples, which are drawn from the product documentation, illustrate the point.

1. In the product documentation, before Version 7.5, the scan was used as a step in tasks to change the installation of IBM WebSphere MQ. The purpose was to detect when all queue managers had ended. In Version 7.5, the tasks use the **dspmq** command to detect when all queue managers associated with a specific installation have ended.
2. In the product documentation, a process scan is used to monitor starting a queue manager in a high availability cluster. Another script is used to stop a queue manager. In the script to stop a queue manager, if the queue manager does not end within a period of time, the list of processes is piped into a **kill -9** command. In both these cases, the scan filters on the queue manager name, and is unaffected by the change to multiple installations.

Related information:

Stopping a queue manager under the control of a high availability (HA) cluster
In most cases, you can use a shell script to stop a queue manager. Examples of suitable shell scripts are given here. You can tailor these to your needs and use them to stop the queue manager under control of your HA cluster.

Monitoring a queue manager
It is usual to provide a way for the high availability (HA) cluster to monitor the state of the queue manager periodically. In most cases, you can use a shell script for this. Examples of suitable shell scripts are given here. You can tailor these scripts to your needs and use them to make additional monitoring checks specific to your environment.

UNIX and Linux: `/usr` symbolic links removed

On all UNIX and Linux platforms, the links from the `/usr` file system are no longer made automatically. In order to take advantage of these links, you must set an installation as the primary installation or set the links up manually.

In previous releases the installation of IBM WebSphere MQ on UNIX and Linux created the symbolic links shown in Table 72 on page 592. In Version 7.5, these links are not created. You must run **setmqinst** to create a primary installation containing symbolic links. No symbolic links are created in other installations.

Table 72. Default symbolic links created in releases before Version 7.1

Symbolic link from	To
/usr/bin/amq...	/opt/mqm/bin/amq...
/usr/lib/amq...	/opt/mqm/lib/amq...
/usr/include/cmq...	/opt/mqm/inc/cmq...
/usr/share/man/...	/opt/mqm/man/...

Only a subset of those links created with previous releases are now made, see External library and control command links to primary installation on UNIX and Linux

For more information about choosing whether to have a primary installation, see Choosing a primary installation

Related tasks:

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

setmqinst

Windows: amqmsrvn.exe process removed

The amqmsrvn.exe DCOM process was replaced by a Windows service, amqsvc.exe, in Version 7.1. This change is unlikely to cause any problems. However, you might have to make some changes. You might have configured the user that runs the IBM WebSphere MQ Windows service MQSeriesServices without the user right to "Log on as a service". Alternatively, the user might not have "List Folder" privilege on all the subdirectories from the root of the drive to the location of the service amqsvc.exe.

If you omitted the "Log on as a service" user privilege, or one of the subdirectories under which IBM WebSphere MQ is installed does not grant the "List Folder" privilege to the user, the MQ_*InstallationName* IBM WebSphere MQ Windows services in Version 7.5 fails to start.

Diagnosing the problem

If the service fails to start, Windows event messages are generated:

- If you did not give the user the "Log on as a service" user privilege, the Windows Service Control Manager adds an event: `7038: The user has not been granted the requested logon type`. The **strmqsvc** command reports error 1069.
- If you did not give the user the "List Folder" privilege, the Windows Service Control Manager adds an event: `7009: Timed out waiting for the service to connect`. The **strmqsvc** command reports error 1053.

If the "Prepare WebSphere MQ" wizard encounters a failure when validating the security credentials of the user performing an installation, an error is returned: `WebSphere MQ is not correctly configured for Windows domain users`. This error indicates that the service failed to start.

Resolution

To resolve this problem:

- Check that the user has the "Log on as a service" user privilege
- Check that the user is a member of the local mqm group

- Check that the local mqm group has “List Folder” privilege on each subdirectory in the path to the service `amqsvc.exe`.

Related reference:

“Windows: “Logon as a service” required” on page 594
The user ID that runs the IBM WebSphere MQ Windows service must have the user authority to “Logon as a service”. If the user ID does not have the authority to run the service, the service does not start and it returns an error in the Windows system event log. Typically you will have run the Prepare IBM WebSphere MQ wizard, and set up the user ID correctly. Only if you have configured the user ID manually is it possible that you might have a problem in Version 7.5.

Windows: IgnoredErrorCodes registry key

The registry key used to specify error codes that you do not want written to the Windows Application Event Log has changed.

The contents of this registry key are not automatically migrated. If you want to continue to ignore specific error codes, you must manually migrate the registry key.

Previously, the key was in the following location:

```
HKLM\Software\IBM\MQSeries\CurrentVersion\IgnoredErrorCodes
```

The key is now in the following location:

```
HKLM\Software\IBM\WebSphere MQ\Installation\MQ_INSTALLATION_NAME\IgnoredErrorCodes
```

where *MQ_INSTALLATION_NAME* is the installation name associated with a particular installation of IBM WebSphere MQ.

Related information:

Ignoring error codes under Windows systems

Windows: Installation and infrastructure information

The location of Windows installation and infrastructure information has changed.

A top-level string value, `WorkPath`, in the `HKLM\SOFTWARE\IBM\WebSphere MQ` key, stores the location of the product data directory which is shared between all installations. The first installation on a machine specifies it, subsequent installations pick up the same location from this key.

Other information previously stored in the registry on Windows is now stored in .ini files.

Related information:

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Windows: Local queue performance monitoring

In IBM WebSphere MQ for Windows Version 7.5 it is no longer possible to monitor local queues using the Windows performance monitor.

Use the performance monitoring commands, which are common to all platforms, provided by IBM WebSphere MQ.

Related information:

Real-time monitoring

Windows: “Logon as a service” required

The user ID that runs the IBM WebSphere MQ Windows service must have the user authority to “Logon as a service”. If the user ID does not have the authority to run the service, the service does not start and it returns an error in the Windows system event log. Typically you will have run the Prepare IBM WebSphere MQ wizard, and set up the user ID correctly. Only if you have configured the user ID manually is it possible that you might have a problem in Version 7.5.

You have always been required to give the user ID that you configure to run IBM WebSphere MQ the user authority to “Logon as a service”. If you run the Prepare IBM WebSphere MQ wizard, it creates a user ID with this authority. Alternatively, it ensures that a user ID you provide has this authority.

It is possible that you ran IBM WebSphere MQ in earlier releases with a user ID that did not have the “Logon as a service” authority. You might have used it to configure the IBM WebSphere MQ Windows service MQSeriesServices, without any problems. If you run a IBM WebSphere MQ Windows service in Version 7.5 with the same user ID that does not have the “Logon as a service” authority, the service does not start.

The IBM WebSphere MQ Windows service MQSeriesServices, with the display name `IBM MQSeries`, changed in Version 7.1. A single IBM WebSphere MQ Windows service per server is no longer sufficient. A IBM WebSphere MQ Windows service per installation is required. Each service is named `MQ_`*`InstallationName`*, and has a display name `IBM WebSphere MQ (`*`InstallationName`*`)`. The change, which is necessary to run multiple installations of IBM WebSphere MQ, has prevented IBM WebSphere MQ running the service under a single specific user ID. In Version 7.5, a `MQ_`*`InstallationName`* service must run as a service.

The consequence is a user ID that is configured to run the Windows service `MQ_`*`InstallationName`* must be configured to “Logon as a service”. If the user ID is not configured correctly, errors are returned in the Windows system event log.

Many installations on earlier releases, and installations from Version 7.1 onwards, configure IBM WebSphere MQ with the Prepare IBM WebSphere MQ wizard. The wizard sets up the user ID with the “Logon as a service” authority and configures the IBM WebSphere MQ Windows service with this user ID. Only if, in previous releases, you have configured MQSeriesServices with another user ID that you configured manually, might you have this migration problem to fix.

Windows: MSCS restriction with multiple installations

When you install or upgrade to IBM WebSphere MQ Version 7.5, the first IBM WebSphere MQ installation on the server is the only one that can be used with Microsoft Cluster Server (MSCS). No other installations on the server can be used with MSCS. This restriction limits the use of MSCS with multiple IBM WebSphere MQ installations.

When you run the **`haregtyp`** command it defines the first IBM WebSphere MQ to be installed as an MSCS resource type; see WebSphere MQ MSCS support utility programs. The implications are as follows:

1. You must associate queue managers that are participating in an MSCS cluster with the first installation on the server.
2. Setting the primary installation has no effect on which installation is associated with the MSCS cluster.
3. If you are upgrading from Version 7.0.1 to Version 7.5, you must follow the single-stage migration scenario; see “UNIX, Linux, and Windows: Single-stage migration from Version 7.0.1, or later, to the latest version” on page 394.

Windows Migration of registry information

Before Version 7.1 all IBM WebSphere MQ configuration information, and most queue manager configuration information, was stored in the Windows registry. From Version 7.1 onwards, all queue manager configuration information (for example, `mqs.ini`, `qmstatus.ini`, and `qm.ini`) is stored in files; the same files as in UNIX and Linux platforms.

The change does not affect the operation of existing applications or queue managers, but it does affect any administrative procedures and scripts that reference the registry.

In Version 7.0.1, to support multi-instance (high availability) queue managers, the queue manager configuration information of some queue managers is stored in `qm.ini` and `qmstatus.ini` rather than in the registry.

The Version 7.0.1 configuration information is accessed from other installations. You must stop all the queue managers and IBM WebSphere MQ applications running on the server to release any locks.

The transfer of configuration data from the registry to files is automatic. It takes place when:

- You migrate an existing Windows system to Version 7.1or later, or
- Version 7.0.1 is uninstalled from a server that has a Version 7.1, or later, installation.

As a consequence, after uninstallation of Version 7.0.1 on a multi-installation server, it is difficult to restore a Version 7.0.1 installation to run any queue managers that you want to restore to the `701` command level:

1. You cannot reinstall Version 7.0.1 on the server. You must run the queue managers on a different server.
2. When you transfer the queue manager data to another server, with Version 7.0.1 installed, you must create the correct registry configuration entries. The entries are not available to copy from the registry on the multi-installation server. Back up the registry entries before uninstalling Version 7.0.1.

Related tasks:

"UNIX, Linux, and Windows: Side-by-side migration from version 7.0.1, or later, to the latest version" on page 401
Side-by-side migration is the term used to describe installing a new version of IBM WebSphere MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM WebSphere MQ. They remain associated with the older version of IBM WebSphere MQ. When you decide to migrate queue managers to the new version of IBM WebSphere MQ, you stop all queue managers, uninstall the old version , and migrate them all to the new version of IBM WebSphere MQ.

"UNIX, Linux, and Windows: Multi-stage migration from version 7.0.1.6, or later, to version 7.5" on page 409
Multi-stage migration is the term used to describe running a new version of IBM WebSphere MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

Related information:

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Windows: Relocation of the mqclient.ini file

In IBM WebSphere MQ for Windows Version 7.5 the `mqclient.ini` file has moved from FilePath to WorkPath. This is similar to the model already used on UNIX and Linux systems.

For users who supply separate file and work paths you will see a change in behavior. You have an additional step to perform when you choose to uninstall IBM WebSphere MQ Version 7.0 before installing IBM WebSphere MQ Version 7.5. Before uninstalling IBM WebSphere MQ Version 7.0, you must copy `mqclient.ini` directly to the Config directory in your data path so that it can be picked up by the IBM WebSphere MQ Version 7.5 installation.

Windows: Task manager interpretation

The interpretation of the processes listed by the Windows Task Manager has changed. The results can show IBM WebSphere MQ processes running for multiple installations on a server. Before Version 7.5, the process list identified IBM WebSphere MQ processes running on only a single installation of IBM WebSphere MQ on a Windows server.

The implications of this change depend on how the results are qualified and interpreted, and how the list of processes is used. The change affects you, only if you start to run multiple installations on a single server. If you have incorporated the list of IBM WebSphere MQ processes into administrative scripts or manual procedures, you must review the usage.

Related information:

Stopping queue managers in WebSphere MQ for Windows

Windows: IBM WebSphere MQ Active Directory Services Interface

The IBM WebSphere MQ Active Directory Services Interface is no longer available.

If your application uses the IBM WebSphere MQ Active Directory Services Interface, you must rewrite your application to use Programmable Command Formats.

Related information:

Introduction to Programmable Command Formats

Changes from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5

All IBM WebSphere MQ changes that affect the migration of a queue manager from Version 6.0 to Version 7.5 are listed.

New capabilities that do not affect existing IBM WebSphere MQ applications are not listed.

On platforms other than z/OS you cannot reverse queue manager migration to remove the effect of changes.

Click on a link to read the details about the change.

Apache Axis shipped with IBM WebSphere MQ updated from version 1.1 to 1.4

The level of Apache Axis shipped with IBM WebSphere MQ is updated to version 1.4 from a patched version of Axis 1.1.

Continue to use the Axis 1.1 JAR file with applications that were built with WebSphere MQ version 6.0. Build new applications that use the IBM WebSphere MQ web transport for SOAP with the Axis 1.4 JAR file.

If you are writing clients that use W3C SOAP over JMS, download Axis 1.4.1 from Apache Axis. Build applications that use W3C SOAP over JMS with the Axis 1.4.1 JAR file.

Note: The naming of versions and releases used by Axis causes confusion. Typically, Axis 1.4 refers to the JAX-RPC implementation, and Axis2 to the JAX-WS implementation.

Axis 1.4 is a version level. If you search for Axis 1.4 on the internet, you are taken to http://ws.apache.org/axis/. The page contains a list of preceding versions of Axis (1.2, 1.3) and the April 22, 2006, final release of Axis 1.4. There are later releases of Axis 1.4, that fix bugs, but they are all known as Axis 1.4. It is one of these bug fix releases that is shipped with WebSphere MQ. For Axis 1.4, use the version of `axis.jar` that is shipped with WebSphere MQ rather than the one obtainable from http://ws.apache.org/axis/.

The Axis website also refers to Axis 1.1 to refer to all the versions of what is more typically called Axis 1.4. Axis 1.2 is used to refer to what is typically called Axis2.

Axis 1.5 is not a later release of Axis 1.4, it is an Axis2 release. If you search for Axis 1.5 you are directed to http://ws.apache.org/axis2/. http://ws.apache.org/axis2/download.cgi contains a list of release versions of Axis2, labeled 0.9 to 1.5.1 (and including, confusingly version 1.4). The release version of Axis2 to use with WebSphere MQ transport for SOAP is 1.4.1. Download Axis2 1.4.1 from http://ws.apache.org/axis2/download/1_4_1/download.cgi.

Related information:

Installing WebSphere MQ web transport for SOAP

Change in behavior of the `endmqm` command

Issuing an **`endmqm`** command and **`dspmq`** command immediately after each other might return misleading status.

When issuing an **`endmqm -c`** or **`endmqm -w`** command, in the unlikely event that a dspmq command is issued in the small timeframe between the applications disconnecting and the queue manager actually stopping, the dspmq command might report the status as `Ending immediately`, even though a controlled shutdown is actually happening.

Related information:

endmqm

dspmq

Changes to cluster error recovery on servers other than z/OS

Before Version 7.5, if a queue manager detected a problem with the local repository manager managing a cluster, it updated the error log. In some cases, it then stopped managing clusters. The queue manager continued to exchange applications messages with a cluster, relying on its increasingly out of date cache of cluster definitions. From Version 7.5 onwards, the queue manager reruns operations that caused problems, until the problems are resolved. If, after five days, the problems are not resolved, the queue manager shuts down to prevent the cache becoming more out of date. As the cache becomes more out of date, it causes a greater number of problems. The changed behavior regarding cluster errors in Version 7.5 does not apply to z/OS.

Every aspect of cluster management is handled for a queue manager by the local repository manager process, `amqrrmfa`. The process runs on all queue managers, even if there are no cluster definitions.

Before Version 7.5, if the queue manager detected a problem in the local repository manager, it stopped the repository manager after a short interval. The queue manager kept running, processing application messages and requests to open queues, and publish or subscribe to topics.

With the repository manager stopped, the cache of cluster definitions available to the queue manager became more out of date. Over time, messages were routed to the wrong destination, and applications failed. Applications failed attempting to open cluster queues or publication topics that had not been propagated to the local queue manager.

Unless an administrator checked for repository messages in the error log, the administrator might not realize the cluster configuration had problems. If the failure was not recognized over an even longer time, and the queue manager did not renew its cluster membership, even more problems occurred. The instability affected all queue managers in the cluster, and the cluster appeared unstable.

From Version 7.5 onwards, IBM WebSphere MQ takes a different approach to cluster error handling. Rather than stop the repository manager and keep going without it, the repository manager reruns failed operations. If the queue manager detects a problem with the repository manager, it follows one of two courses of action.

1. If the error does not compromise the operation of the queue manager, the queue manager writes a message to the error log. It reruns the failed operation every 10 minutes until the operation succeeds. By default, you have five days to deal with the error; failing which, the queue manager writes a message to the error log, and shuts down. You can postpone the five day shutdown.
2. If the error compromises the operation of the queue manager, the queue manager writes a message to the error log, and shuts down immediately.

An error that compromises the operation of the queue manager is an error that the queue manager has not been able to diagnose, or an error that might have unforeseeable consequences. This type of error often results in the queue manager writing an FFST file. Errors that compromise the operation of the queue manager might be caused by a bug in IBM WebSphere MQ, or by an administrator, or a program, doing something unexpected, such as ending a IBM WebSphere MQ process.

The point of the change in error recovery behavior is to limit the time the queue manager continues to run with a growing number of inconsistent cluster definitions. As the number of inconsistencies in cluster definitions grows, the chance of abnormal application behavior grows with it.

The default choice of shutting down the queue manager after five days is a compromise between limiting the number of inconsistencies and keeping the queue manager available until the problems are detected and resolved.

You can extend the time before the queue manager shuts down indefinitely, while you fix the problem or wait for a planned queue manager shutdown. The five-day stay keeps the queue manager running through a long weekend, giving you time to react to any problems or prolong the time before restarting the queue manager.

Corrective actions

You have a choice of actions to deal with the problems of cluster error recovery. The first choice is to monitor and fix the problem, the second to monitor and postpone fixing the problem, and the final choice is to continue to manage cluster error recovery as in releases before Version 7.5.

1. Monitor the queue manager error log for the error messages `AMQ9448` and `AMQ5008`, and fix the problem.

   `AMQ9448` indicates that the repository manager has returned an error after running a command. This error marks the start of trying the command again every 10 minutes, and eventually stopping the queue manager after five days, unless you postpone the shutdown.

   `AMQ5008` indicates that the queue manager was stopped because a IBM WebSphere MQ process is missing. `AMQ5008` results from the repository manager stopping after five days. If the repository manager stops, the queue manager stops.
2. Monitor the queue manager error log for the error message `AMQ9448`, and postpone fixing the problem.

   If you disable getting messages from `SYSTEM.CLUSTER.COMMAND.QUEUE`, the repository manager stops trying to run commands, and continues indefinitely without processing any work. However, any handles that the repository manager holds to queues are released. Because the repository manager does not stop, the queue manager is not stopped after five days.

   Run an MQSC command to disable getting messages from `SYSTEM.CLUSTER.COMMAND.QUEUE`:

   ```
   ALTER QLOCAL(SYSTEM.CLUSTER.COMMAND.QUEUE) GET(DISABLED)
   ```

To resume receiving messages from SYSTEM.CLUSTER.COMMAND.QUEUE run an MQSC command:

```
ALTER QLOCAL(SYSTEM.CLUSTER.COMMAND.QUEUE) GET(ENABLED)
```

3. Revert the queue manager to the same cluster error recovery behavior as before Version 7.5.

   You can set a queue manager tuning parameter to keep the queue manager running if the repository manager stops.

   The tuning parameter is TolerateRepositoryFailure, in the TuningParameters stanza of the qm.ini file. To prevent the queue manager stopping, if the repository manager stops, set TolerateRepositoryFailure to TRUE; see Figure 94.

   Restart the queue manager to enable the TolerateRepositoryFailure option.

   If a cluster error has occurred that prevents the repository manager starting successfully, and hence the queue manager from starting, set TolerateRepositoryFailure to TRUE to start the queue manager without the repository manager.

Special consideration

Before Version 7.5, some administrators managing queue managers that were not part of a cluster stopped the amqrrmfa process. Stopping amqrrmfa did not affect the queue manager.

Stopping amqrrmfa in Version 7.5 causes the queue manager to stop, because it is regarded as a queue manager failure. You must not stop the amqrrmfa process in Version 7.5, unless you set the queue manager tuning parameter, TolerateRepositoryFailure.

Example

```
TuningParameters:
    TolerateRepositoryFailure=TRUE
```

Figure 94. Set TolerateRepositoryFailure to TRUE in qm.ini

Related information:

Queue manager configuration files, qm.ini
A queue manager configuration file, qm.ini, contains information relevant to a specific queue manager.

AMQ9448

MQI client: Changes

A IBM WebSphere MQ MQI client can connect to queue managers running on earlier or later releases of IBM WebSphere MQ without recompilation or relinking. The behavior of a client application might change. Changes that are specific to clients are changes to the client definition table, to the default settings of client and server-connection channels, and to the behavior of MQPUT1. Clients might also be affected by other changes to IBM WebSphere MQ.

MQI client: Client Channel Definition Table (CCDT):

The client channel definition table has changed from Version 6.0 to Version 7.5. Existing clients must continue to use existing CCDTs. To use a Version 7.5 CCDT, you must update the client.

You can connect a IBM WebSphere MQ MQI client application to any level of queue manager. If the client uses a CCDT, it must use a CCDT built by the same or earlier version of queue manager. If the client connects without using a CCDT, this restriction does not apply.

In a common migration scenario, if you upgrade a Version 6.0 queue manager to Version 7.5, and you do not create new CCDTs for its clients, the clients connect to the Version 7.5 queue manager without any changes being required. Client behavior might change as a result of changes to the queue manager.

Another common migration scenario is to update some queue managers and some clients to Version 7.5, leaving some queue managers and clients at Version 6.0. In this scenario, you want to update the CCDT for Version 7.5 IBM WebSphere MQ MQI client connected to Version 7.5 queue managers to Version 7.5, so that the clients can exploit Version 7.5 function fully. The new clients also connect to Version 6.0 queue managers. Existing clients connect to both Version 6.0 and Version 7.5 queue managers. In order that Version 7.5 clients can exploit new Version 7.5 function, you must deploy a Version 7.5 CCDT to Version 7.5 clients. Version 6.0 clients must continue to use the Version 6.0 CCDT. Both sets of clients can connect to both sets of queue managers, regardless of the CCDT they are using.

If the client is a IBM WebSphere MQ MQI client, the version of the IBM WebSphere MQ MQI client libraries linked to by the client must be the same or greater than the version of the queue manager that was used to build the CCDT. If the client is a Java or JMS client, then the client must be built with versions of the WebSphere MQ JAR files that are the same or greater than the queue manager that was used to build the CCDT.

To upgrade a Version 6.0 IBM WebSphere MQ MQI client to use a Version 7.5 CCDT, you must upgrade the IBM WebSphere MQ MQI client installation to Version 7.5. Unless you decide to do so for other reasons, do not rebuild the client application.

To upgrade a Version 6.0 Java or JMS client to use a Version 7.5 CCDT, redeploy the WebSphere MQ JAR files to the client workstation. You do not need to rebuild the Java or JMS client with the new JAR files.

MQI client: Client configuration stanzas moved into a new configuration file:

Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

Moving client configuration information affects existing settings; for example:

- Set the TCP `KeepAlive` attribute for client connections in `mqclient.ini`; for example:

  ```
  TCP:
     KeepAlive = Yes
  ```

  An existing setting in `qm.ini` or in the Windows registry is ignored.
- Set `ClientExitPath` in `mqclient.ini`; for example:

  ```
  ClientExitPath:
     ExitsDefaultPath=/var/mqm/exits
     ExitsDefaultPath64=/var/mqm/exits64
  ```

  An existing setting in `mqs.ini` or in the Windows registry is moved to the client configuration file when you upgrade the client. If you add values to `mqs.ini` or the Windows registry, they are ignored.
- Set `JavaExitsClasspath` in `mqclient.ini`.

  Do not continue to use the Java system property`com.ibm.mq.exitClasspath`. Existing settings continue to work, but they are deprecated. The setting in `mqclient.ini` has precedence over the Java system property.

Related information:

The WebSphere MQ classes for JMS configuration file

Assigning channel exits for WebSphere MQ classes for JMS

WebSphere MQ client configuration file

Configure your clients using attributes in a text file. These attributes can be overridden by environment variables or in other platform-specific ways.

MQI client: Default behavior of client-connection and server-connection:

The default settings for client and server connection channels have changed to use the new shared conversations capability. The change has an impact on performance, and on the behavior of heartbeats and channels exits.

From Version 7.0, the default for client and server connections is to share an MQI channel. Each channel is defined with a default of 10 threads to run up to 10 client conversations per channel instance. Before Version 7.0, each conversation was allocated to a different channel instance. The change might cause migration problems for existing client applications. You can restore Version 6.0 behavior of a server or client connection channel by setting the channel attribute, **SHARECNV**, to 0.

Note: If you set **SHARECNV** to 1, rather than 0, each conversation is allocated a separate channel instance. However, the channel behaves like a new channel. For example, heartbeats flow in each direction at any time, and the channel supports the following features:

- Administrator stop-quiesce
- Unrestricted heartbeats
- Read-ahead
- Asynchronous-consume by client applications

You can set the MQCONNX option, MQCNO_NO_CONV_SHARING and connect the application to a channel with **SHARECNV** set to a value greater than 1. The result is the same as connecting the application to a channel with **SHARECNV** set to 1. If you connect the application to a channel with **SHARECNV** set to 0, the connection behaves like a version 6.0 MQI channel.

Performance

Processing of messages on channels that use the default configuration of 10 shared conversations is, on average, 15% slower than on Version 6.0. Version 6.0 channels do not share conversations.

- Established client applications that do not use new features can use channels without shared conversations. Set **SHARECNV** to 0.
- New client applications can mitigate the performance degradation incurred by sharing, by taking advantage of features like read-ahead and client asynchronous-consume. These features are only available with MQI channels configured to use shared conversations.

On an MQI channel instance that is sharing conversations, all of the conversations on a socket are received by the same thread. By using a single queue manager thread, a high **SHARECNV** value reduces queue manager thread usage, and generally improves performance. However, if the conversations sharing a socket are all busy, the conversational threads contend with one another to use the receiving thread. The contention cause delays, and in this situation a lower **SHARECNV** value is better. A **SHARECNV** value of 1 eliminates contention to use the receiving thread and makes new features available.

Heartbeats

Heartbeats can now flow across the channel at any time in either direction. The Version 6.0 behavior is for heartbeats to flow only when an MQGET call is waiting. Restore the Version 6.0 behavior by setting **SHARECNV** to 0.

Channel exits

The behavior of a client or server connection channel exit changes when the channel is sharing conversations. It is unlikely that the change affects the actual behavior of existing exits; but it could. The change is as follows:

Updating `MQCD` when `SharingConversations` field is set to `TRUE` does not affect the way the channel runs. Only alterations made when `MQCXP SharingConversations` field is set to `FALSE`, on an MQXR_INIT call, change channel behavior.

- Send or receive exits can alter the `MQCD` structure on an MQXR_INIT call. The effect of these exits differs, depending on whether the conversation is shared with other conversations on the same channel:
  - If the `MQCXP SharingConversations` field passed to the exit instance is set to `FALSE`, this exit instance is the first, or only, conversation on the channel instance. No other exit can be changing the `MQCD` at the same time, and changes that are made to the `MQCD` can affect the way that the channel runs.
  - If the `MQCXP SharingConversations` field passed to the exit instance is set to `TRUE`, this exit instance is a subsequent conversation. It is sharing the channel instance with other conversations. Changes made to the `MQCD` in the exit instance are retained in the `MQCD` but do not affect the way the channel runs.
- Send, receive, and security exit instances can alter the `MQCD`, when the `MQCXP SharingConversations` field is set to `TRUE`. Exit instances on other conversations might be changing the `MQCD` at the same time. Updates written by one exit instance can be overwritten by another instance. It might be necessary to serialize access to the `MQCD` across these different exit instances to maintain the consistency of fields in `MQCD`.

Related information:

Using sharing conversations

HeartbeatInterval (MQLONG)

SharingConversations (MQLONG)

ALTER CHANNEL

Channel-exit programs for MQI channels
Three types of channel exit are available to the WebSphere MQ MQI client environment on UNIX, Linux and Windows systems.

Using read ahead

Stopping channels

The Asynchronous consume sample program

MQI client: MQPUT1 sync point behavior change:

An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

In the scenario, "Example call sequence that demonstrates the change in behavior" on page 603, an MQPUT1 call can succeed where it failed in Version 6.0. The result occurs when all the following conditions are met:

- Both client and queue manager are later than Version 6.0.
- The application program is connected to the queue manager as a client application
- `MQPMO_SYNCPOINT` is set in the Put Message Options structure, `MQPMO`

You can make the IBM WebSphere MQ MQI client behave like Version 6.0. Set **`Put1DefaultAlwaysSync`** to `YES` in the **`CHANNELS`** stanza of the client configuration file; see Figure 95 on page 603.

```
Channels:
    Put1DefaultAlwaysSync=YES
```

Figure 95. Add **`Put1DefaultAlwaysSync`** *to `mqclient.ini`*

Example call sequence that demonstrates the change in behavior

1. MQCONN to queue manager from a WebSphere MQ MQI client application.
2. MQPUT1 to a nonexistent queue with the `MQPMO_SYNCPOINT` option
3. MQDISC

In IBM WebSphere MQ Version 6.0 the MQPUT1 call ends with `MQCC_FAILED` and `MQRC_UNKNOWN_OBJECT_NAME(2085)`. Running with a client and server later than Version 6.0, the MQPUT1 call ends with `MQRC_NONE` and `MQCC_OK`.

Related information:

CHANNELS stanza of the client configuration file
Use the CHANNELS stanza to specify information about client channels.

Channel authentication

When you migrate a queue manager to Version 7.5, channel authentication using channel authentication records is disabled. Channels continue to work as before. If you create a queue manager in Version 7.5, channel authentication using channel authentication records is enabled, but with minimal additional checking. Some channels might fail to start.

Migrated queue managers

Channel authentication is disabled for migrated queue managers.

To start using channel authentication records you must run this MQSC command:

```
ALTER QMGR CHLAUTH(ENABLED)
```

New queue managers

Channel authentication is enabled for new queue managers.

You want to connect existing queue managers or IBM WebSphere MQ MQI client applications to a newly created queue manager. Most connections work without specifying any channel authentication records. The following exceptions are to prevent privileged access to the queue manager, and access to system channels.

1. Privileged user IDs asserted by a client-connection channel are blocked by means of the special value `*MQADMIN`.

   ```
   SET CHLAUTH('*') TYPE(BLOCKUSER) USERLIST('*MQADMIN') +
   DESCR('Default rule to disallow privileged users')
   ```

2. Except for the channel used by IBM WebSphere MQ Explorer, all `SYSTEM.*` channels are blocked.

   ```
   SET CHLAUTH('SYSTEM.*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) +
   DESCR('Default rule to disable all SYSTEM channels')

   SET CHLAUTH(SYSTEM.ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(CHANNEL) +
   DESCR('Default rule to allow MQ Explorer access')
   ```

Note: This behaviour is default for all new IBM WebSphere MQ Version 7.5 queue managers on startup.

If you must work around the exceptions, you can run an MQSC command to add in more rules to allow channels blocked by the default rules to connect, or disable channel authentication checking:

```
ALTER QMGR CHLAUTH(DISABLED)
```

Related information:

Channel authentication records

Command level changed to 750

The command level on platforms other than z/OS and IBM i changes to 750 in Version 7.5. z/OS and IBM i are at command level 710.

Related information:

MQCMDL_* (Command Levels)

CommandLevel (MQLONG)

Connect to multiple queue managers and use MQCNO_FASTPATH_BINDING

Applications that connect to queue managers using the MQCNO_FASTPATH_BINDING binding option might fail with an error and reason code MQRC_FASTPATH_NOT_AVAILABLE.

An application can connect to multiple queue managers from the same process. In releases earlier than Version 7.5, an application can set any one of the connections to MQCNO_FASTPATH_BINDING. In Version 7.5, only the first connection can be set to MQCNO_FASTPATH_BINDING. See "Fast path" for the complete set of rules.

To assist with migration, you can set a new environment variable, AMQ_SINGLE_INSTALLATION. The variable reinstates the same behavior as in earlier releases, but prevents an application connecting to queue managers associated with other installations in the same process.

Fast path

On a server with multiple installations, applications using a fast path connection to IBM WebSphere MQ Version 7.1 or later must follow these rules:

1. The queue manager must be associated with the same installation as the one from which the application loaded the IBM WebSphere MQ run time libraries. The application must not use a fast path connection to a queue manager associated with a different installation. An attempt to make the connection results in an error, and reason code MQRC_INSTALLATION_MISMATCH.
2. Connecting non-fast path to a queue manager associated with the same installation as the one from which the application has loaded the IBM WebSphere MQ run time libraries prevents the application connecting fast path, unless either of these conditions are true:
   - The application makes its first connection to a queue manager associated with the same installation a fast path connection.
   - The environment variable, AMQ_SINGLE_INSTALLATION is set.
3. Connecting non-fast path to a queue manager associated with a Version 7.1 or later installation, has no effect on whether an application can connect fast path.
4. You cannot combine connecting to a queue manager associated with a Version 7.0.1 installation and connecting fast path to a queue manager associated with a Version 7.1, or later installation.

With AMQ_SINGLE_INSTALLATION set, you can make any connection to a queue manager a fast path connection. Otherwise almost the same restrictions apply:

- The installation must be the same one from which the IBM WebSphere MQ run time libraries were loaded.
- Every connection on the same process must be to the same installation. If you attempt to connect to a queue manager associated with a different installation, the connection fails with reason code MQRC_INSTALLATION_MISMATCH. Note that with AMQ_SINGLE_INSTALLATION set, this restriction applies to all connections, not only fast path connections.

- Only connect one queue manager with fast path connections.

Related information:

Binding options

2587 (0A1B) (RC2587): MQRC_HMSG_NOT_AVAILABLE

2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

Custom scripts

In IBM WebSphere MQ for Windows Version 7.5 custom scripts to install packages can fail as they have now been renamed.

Custom scripts to install IBM WebSphere MQ can be incomplete as they have been added or removed.

For a detailed explanation of each component, see Choosing what to install and select your platform.

Changes to data types

A number of data types have changed between IBM WebSphere MQ Version 6.0 and IBM WebSphere MQ Version 7.5, and new data types have been added. This topic lists the changes for data types that have a new current version since IBM WebSphere MQ Version 6.0.

The current version of a data type is incremented if the length of a data type is extended by adding new fields. The addition of new constants to the values that can be set in a data type does not result in a change to the current version value.

Table 73 lists the data types that have new versions. Click on the links to read about the new fields.

Table 73. New fields added to existing data types

Data type	**New version**	**New fields**
Authentication information record	`MQAIR_VERSION_2`	OCSPResponderURL (MQCHAR256)
Get-message options	`MQGMO_VERSION_4`	Reserved2 (MQLONG) MsgHandle (MQHMSG)
Object descriptor	`MQOD_VERSION_4`	ObjectString (MQCHARV) SelectionString (MQCHARV) ResObjectString (MQCHARV) ResolvedType (MQLONG)
Put-message options	`MQPMO_VERSION_3`	OriginalMsgHandle (MQHMSG) NewMsgHandle (MQHMSG) Action (MQLONG) PubLevel (MQLONG)
Channel definition	`MQCD_VERSION_9`	SharingConversations PropertyControl MaxInstances MaxInstancesPerClient ClientChannelWeight ConnectionAffinity
	`MQCD_VERSION_10`	BatchDataLimit (MQLONG) DefReconnect (MQLONG) UseDLQ (MQLONG)
Exit parameter structure	`MQAXP_VERSION_2`	ExitMsgHandle

Table 73. New fields added to existing data types (continued)

Data type	New version	New fields
Channel exit parameter	MQCXP_VERSION_7	Connection handle SharingConversations
	MQCXP_VERSION_8	MCAUserSource (MQLONG) pEntryPoints (PMQIEP)
Cluster workload queue-record structure	MQWQR_VERSION_3	DefPutResponse
Cluster workload exit parameter structure	MQWXP_VERSION_4	pEntryPoints (PMQIEP)

Default transmission queue restriction

The product documentation in previous versions of IBM WebSphere MQ warned about defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In Version 7.5, any attempt to set or use a default transmission queue that is defined as `SYSTEM.CLUSTER.TRANSMIT.QUEUE` results in an error.

In earlier versions of IBM WebSphere MQ no error was reported when defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. MQOPEN or MQPUT1 MQI calls that resulted in referencing the default transmission queue did not return an error. Applications might have continued working and failed later on. The reason for the failure was hard to diagnose.

The change ensures that any attempt to set the default transmission queue to `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, or use a default transmission queue set to `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, is immediately reported as an error.

Related reference:

"MQI and PCF reason code changes" on page 584
Reason codes that have changed and that affect some existing programs, are listed.

Related information:

AMQ8520: The queue name supplied is not valid for DEFXMITQ.

Display channel and cluster status: Switching

From IBM WebSphere MQ Version 7.5, a cluster-sender channel that is switching its configuration to a different cluster transmission queue has a new channel state: Switching.

When migrating from a release before Version 7.5 to Version 7.5 or later, existing application programs are not affected by the new state.

System management programs that monitor channel or cluster status might receive the new state as a result of a inquiry.

The state is set during the short interval while the channel modifies the destination transmission queue that messages are stored on. Before the switching state is set, messages are stored on the previously associated transmission queue. After the switching state, messages are stored on the newly configured transmission queue. The channel enters the switching state if a cluster-sender channel is starting, a configuration change is required, and the conditions for starting the switch are met.

Related information:

Channel states
A channel can be in one of many states at any time. Some states also have substates. From a given state a channel can move into other states.

Cluster transmission queues and cluster-sender channels
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

DISPLAY CHSTATUS

Inquire Channel Status (Response)

Inquire Cluster Queue Manager (Response)

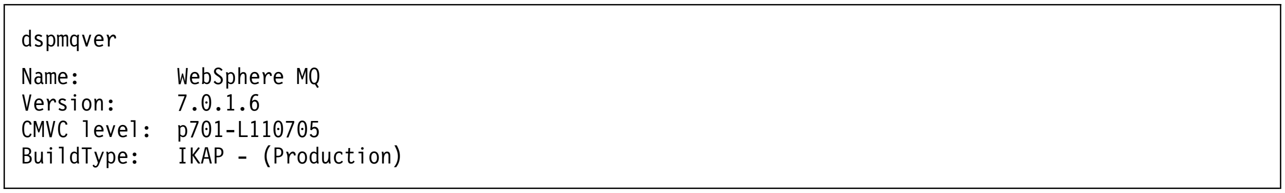
MQCHS_* (Command format Channel Status)

dspmqver

New types of information are displayed by **dspmqver** to support multiple installations. The changes might affect existing administrative scripts you have written to manage IBM WebSphere MQ.

The changes in output from **dspmqver** that might affect existing command scripts that you have written are twofold:

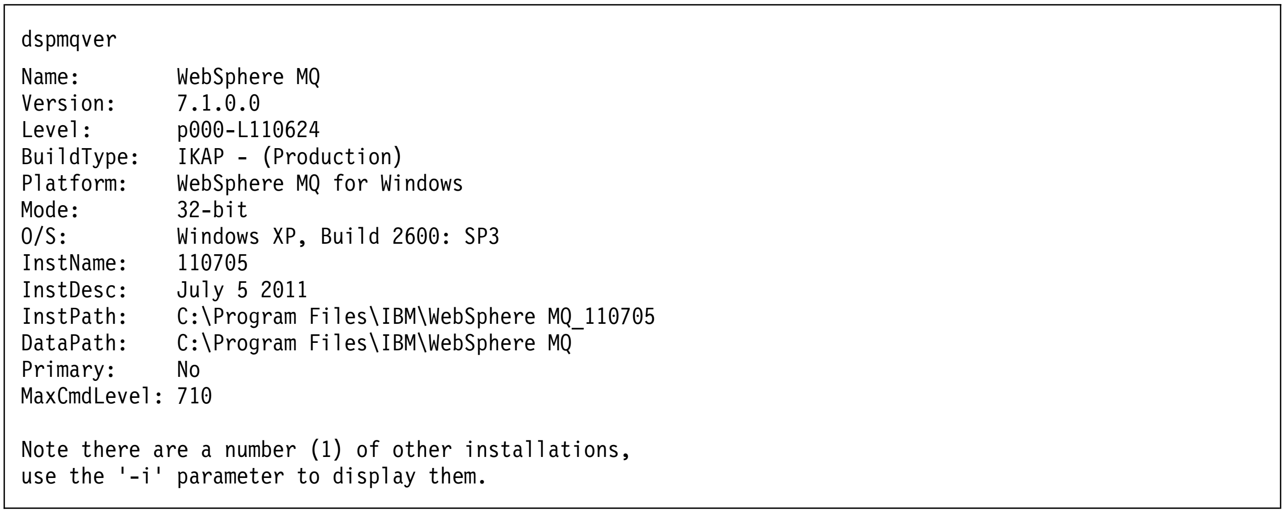
1. Version 7.5 has extra `-f` field options. If you do not specify a `-f` option, output from all the options is displayed. To restrict the output to the same information that was displayed in earlier releases, set the `-f` option to a value that was present in the earlier release. Compare the output for `dspmqver` in Figure 96 and Figure 97 with the output for `dspmqver -f 15` in Figure 98 on page 608.



```
dspmqver

Name:        WebSphere MQ
Version:     7.0.1.6
CMVC level:  p701-L110705
BuildType:   IKAP - (Production)
```

Figure 96. Default **dspmqver** *options in IBM WebSphere MQ Version 7.0.1*



```
dspmqver

Name:        WebSphere MQ
Version:     7.1.0.0
Level:       p000-L110624
BuildType:   IKAP - (Production)
Platform:    WebSphere MQ for Windows
Mode:        32-bit
O/S:         Windows XP, Build 2600: SP3
InstName:    110705
InstDesc:    July 5 2011
InstPath:    C:\Program Files\IBM\WebSphere MQ_110705
DataPath:    C:\Program Files\IBM\WebSphere MQ
Primary:     No
MaxCmdLevel: 710

Note there are a number (1) of other installations,
use the '-i' parameter to display them.
```

Figure 97. Default **dspmqver** *options in IBM WebSphere MQ Version 7.5*

```
dspmqver -f 15

Name:        WebSphere MQ
Version:     7.1.0.0
Level:       p000-L110624
BuildType:   IKAP - (Production)
```

Figure 98. **dspmqver** *with option to make IBM WebSphere MQ Version 7.5 similar to IBM WebSphere MQ Version 7.0.1*

2. The heading of the build level row has changed from `CMVC level:` to `Level:`.

Related information:

dspmqver

Exits and installable services

When migrating to IBM WebSphere MQ Version 7.5 for a distributed platform, if you install IBM WebSphere MQ in a non-default location, you must update your exits and installable services. Data conversion exits generated using the **crtmqcvx** command must be regenerated using the updated command.

When writing new exits and installable services, you do not need to link to any of the following IBM WebSphere MQ libraries:

- mqmzf
- mqm
- mqmvx
- mqmvxd
- mqic
- mqutl

For more information about updating existing exits, and writing exits and installable services that do not link to the libraries, see Writing and compiling exits and installable services.

Fewer IBM WebSphere MQ MQI client log messages

A IBM WebSphere MQ MQI client used to report every failed attempt to connect to a queue manager when processing a connection name list. From Version 7.5, only if the failure occurs with the last connection in the list is a message written to the queue manager error log.

Reporting the last failure and no others, reduces growth of the queue manager error log.

GSKit: Changes from GSKit V7.0 to GSKit V8.0

For distributed platforms, GSKit V8.0 is integrated with IBM WebSphere MQ. In versions of IBM WebSphere MQ prior to Version 7.1, you installed GSKit separately. GSKit V8.0 was included as an alternative to GSKit V7.0 in IBM WebSphere MQ Version 7.0.1; it is now the only version of GSKit provided with IBM WebSphere MQ. Some functions in GSKit V8.0 are different to the functions in GSKit V7.0.

Review the following list of changes.

GSKit: Some FIPS 140-2 compliant channels do not start:

Three CipherSpecs are no longer FIPS 140-2 compliant. If a client or queue manager is configured to require FIPS 140-2 compliance, channels that use the following CipherSpecs do not start after migration.

- FIPS_WITH_DES_CBC_SHA
- FIPS_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_DES_CBC_SHA

To restart a channel, alter the channel definition to use a FIPS 140-2 compliant CipherSpec. Alternatively, configure the queue manager, or the client in the case of a IBM WebSphere MQ MQI client, not to enforce FIPS 140-2 compliance.

Earlier versions of IBM WebSphere MQ enforced an older version of the FIPS 140-2 standard. The following CipherSpecs were considered FIPS 140-2 compliant in earlier versions of IBM WebSphere MQ and are also compliant in this version:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (deprecated)
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA256 (only when AltGSKit version 8 is used with Fix Pack 7.0.1.4 or later)
- TLS_RSA_WITH_AES_256_CBC_SHA256 (only when AltGSKit version 8 is used with Fix Pack 7.0.1.4 or later)

Use these CipherSpecs if you want IBM WebSphere MQ to interoperate in a FIPS 140-2 compliant manner with earlier versions.

Previous IBM WebSphere MQ releases enforced an older version of the FIPS 140-2 standard. The following CipherSpecs were considered FIPS 140-2 compliant by previous IBM WebSphere MQ releases and are also considered compliant by IBM WebSphere MQ:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (deprecated)
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA

Use these CipherSpecs if you need IBM WebSphere MQ version 7.1 to interoperate in a FIPS 140-2 compliant manner with earlier IBM WebSphere MQ releases.

Related information:

AMQ9196

Federal Information Processing Standards (FIPS)

Federal Information Processing Standards

FipsRequired (MQLONG)

MQSSLFIPS
MQSSLFIPS specifies whether only FIPS-certified algorithms are to be used if cryptography is carried out in WebSphere MQ. The values are the same as for the SSLFIPS parameter of the ALTER QMGR command.

Specifying that only FIPS-certified CipherSpecs are used at run time on the MQI client

SSLFIPSRequired (MQLONG)

GSKit: Certificate Common Name (CN) not mandatory:

In GSKit V8.0, the **iKeyman** command accepts any element of the distinguished name (DN), or a form of the subject alternative name (SAN). It does not mandate that you provide it with a common name. In GSKit V7.0, if you create a self-signed certificate using the **iKeyman** command you had to specify a common name.

The implication is that applications searching for a certificate might not able to assume that a certificate has a common name. You might need to review how applications search for certificates, and how applications handle errors involving the common name. Alternatively, you might choose to check that all self-signed certificates are given common names.

Some other certificate tools that you might also be using, do not require a common name. It is therefore likely the change to GSKit is not going to cause you a problem.

Related information:

Distinguished Names

GSKit: Commands renamed:

The command name **gsk7cmd** is replaced with **runmqckm**; **gsk7ikm** is replaced with **strmqikm**, and **gsk7capicmd** is replaced with **runmqakm** . All the commands start the GSKit V8.0 certificate administration tools, and not the GSKit V7.0 tools.

IBM WebSphere MQ Version 7.5 does not use a machine-wide shared installation of GSKit: instead it uses a private GSKit installation in the IBM WebSphere MQ installation directory. Each IBM WebSphere MQ Version 7.5 installation can use a different GSKit version. To display the version number of GSKit embedded in a particular WebSphere MQ installation, run the **dspmqver** command from that installation as shown in the following table:

Table 74. Renamed GSKit commands

Platform	GSKit V7.0 command	GSKit V8.0 command
UNIX and Linux	`gsk7cmd`	`runmqckm`
UNIX and Linux	`gsk7ikm`	`strmqikm`
Windows,UNIX and Linux	`gsk7capicmd`	`runmqakm`
Windows,UNIX and Linux	`gsk7ver`	`dspmqver -p 64 -v`

Note: Do not use the **gsk8ver** command to display the GSKit version number: only the **dspmqver** command will show the correct GSKit version number for IBM WebSphere MQ Version 7.5.

Related information:

runmqckm, and runmqakm commands

Using iKeyman, iKeycmd, runmqakm, and runmqckm

dspmqver

GSKit: The iKeyman commands to insert a certificate do not check that all required CA certificates are present:

The **iKeyman** command in GSKit V8.0 does not validate a certificate when it is inserted into a key repository. **iKeyman** in GSKit V7.0, validated a certificate before it inserted the certificate into a certificate store.

The implication is, that if you create a certificate using the **iKeyman** in GSKit V8.0, all the necessary intermediate and root CA certificates might not be present, or they might have expired; when the certificate is checked it might fail.

Necessary certificates might be missing, or have expired. This can cause SSL and TLS connections to fail with error AMQ9633.

Related information:

Certificate validation and trust policy design on UNIX, Linux and Windows systems

GSKit: Certificate stores created by iKeyman and iKeycmd no longer contain CA certificates:

The **iKeyman** and **iKeycmd** utilities in GSKit V8.0 create a certificate store without adding pre-defined CA certificates to the store. To create a working certificate store, you must now add all the certificates that you require and trust. In GSKit V7.0 **iKeyman** and **iKeycmd** created a certificate store that already contained CA certificates.

Existing data bases created by GSKit V7.0 are unaffected by this change.

Related information:

Adding default CA certificates into an empty key repository, on UNIX, Linux or Windows systems with GSKit version 8.0

GSKit: Import of a duplicate PKCS#12 certificate:

In GSKit V8.0, the **iKeyman** command does not report an attempt to import a duplicate PKCS#12 certificate as an error. In GSKit V7.0, the **iKeyman** command reported an error. In neither version is a duplicate certificate imported.

For GSKIT V8.0, a duplicate certificate is a certificate with the same label and public key.

The implication is that if some of the issuer information is different, but the name and public key are the same, the changes are not imported. The correct way to update a certificate is to use the `-cert -receive` option, which replaces an existing certificate.

gskcapicmd does not allow or ignore duplicates on import in this way.

Related information:

Importing a personal certificate into a key repository on UNIX, Linux or Windows systems

GSKit: Password expiry to key database deprecated:

In GSKit V8.0, the password expiry function in **iKeyman** continues to work the same as in GSKit V7.0, but it might be withdrawn in future versions of GSKit.

Use the file system protection provided with the operating system to protect the key database and password stash file.

Linux: Recompile C++ applications and update run time libraries:

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.

Related tasks:

"Linux: Rebuilding a C++ application" on page 491

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

GSKit: Signature algorithm moved out of settings file:

In GSKit V8.0, the default signature algorithm used when creating self-signed certificates or certificate requests or selected in the creation dialogs is passed as a command-line parameter. In GSKit V7.0, the default signature algorithm was specified in the settings file.

The change has very little effect: it causes a different default signature algorithm to be selected. It does not alter the selection of a signature algorithm.

Related information:

runmqckm and runmqakm options

Creating a self-signed personal certificate on UNIX, Linux, and Windows systems

GSKit: Signed certificate validity period not within signer validity:

In GSKit V8.0, the **iKeyman** command does not check whether the validity period of a resulting certificate is within the validity period of the signed certificate. In GSKit V7.0, **iKeyman** checked that the validity period of the resulting certificate was within the validity period of the signed certificate.

The IETF RFC standards for SSL/TLS allow a certificate whose validity dates extend beyond those of its signer. This change to GSKit brings it into line with those standards. The check is whether the certificate is issued within the validity period of the signer, and not whether it expires within the validity period of the signer.

Related information:

How SSL and TLS provide identification, authentication, confidentiality, and integrity

GSKit: Stricter default file permissions:

The default file permissions set by **runmqckm** and **strmqikm** in IBM WebSphere MQ Version 7.5 on UNIX and Linux are stricter than the permissions that are set by **runmqckm**, **strmqikm**, **gsk7cmd**, and **gsk7ikm** in earlier releases of IBM WebSphere MQ.

The permissions set by **runmqckm** and **strmqikm** in IBM WebSphere MQ Version 7.5 permit only the creator to access the UNIX and Linux SSL/TLS key databases. The **runmqckm**, **strmqikm**, **gsk7cmd**, and **gsk7ikm** tools in earlier releases of IBM WebSphere MQ set world-readable permissions, making the files liable to theft and impersonation attacks.

The permissions set by **gsk7capicmd**, in earlier releases of IBM WebSphere MQ, and **runmqakm** in IBM WebSphere MQ Version 7.5, permit only the creator to access UNIX and Linux SSL/TLS key databases.

The migration of SSL/TLS key databases to Version 7.5 does not alter their access permissions. In many cases, administrators set more restrictive access permissions on these files to overcome the liability to theft and impersonation attacks; these permissions are retained.

The default file permissions set on Windows are unchanged. Continue to tighten up the access permissions on SSL/TLS key database files on Windows after creating the files with **runmqckm** or **strmqikm**.

Related information:

Accessing and securing your key database files on Windows

Accessing and securing your key database files on UNIX and Linux systems

JMS and Java changes

WebSphere MQ classes for JMS has been rewritten to integrate it with IBM WebSphere MQ queue manager internal interfaces. To a large extent, its functional behavior remains the same. Some changes affect the way existing JMS and Java applications behave. The differences are noted in the following topics.

Java MQMessage.resizeBuffer exception change:

In IBM WebSphere MQ Version 6.0, attempting to resize a buffer to a negative size using MQMessage.resizeBuffer in a Java program resulted in a NegativeArraySizeException exception. It now results in an IllegalArgumentException exception.

JMS: Reason code changes:

Some reason codes returned in JMS exceptions have changed. The changes affect `MQRC_Q_MGR_NOT_AVAILABLE` and `MQRC_SSL_INITIALIZATION_ERROR`.

In earlier releases of IBM WebSphere MQ, if a JMS application call fails to connect, it receives an exception with a reason code 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE. In Version 7.5, it can still receive `MQRC_Q_MGR_NOT_AVAILABLE`, or one of the following more specific reason codes.

- 2537 (09E9) (RC2537): MQRC_CHANNEL_NOT_AVAILABLE
- 2538 (09EA) (RC2538): MQRC_HOST_NOT_AVAILABLE
- 2539 (09EB) (RC2539): MQRC_CHANNEL_CONFIG_ERROR
- 2540 (09EC) (RC2540): MQRC_UNKNOWN_CHANNEL_NAME

Similarly, when trying to connect, a JMS application might have received 2393 (0959) (RC2393): MQRC_SSL_INITIALIZATION_ERROR. In Version 7.5, it can still receive `MQRC_SSL_INITIALIZATION_ERROR`, or a more specific reason code, such as 2400 (0960) (RC2400): MQRC_UNSUPPORTED_CIPHER_SUITE, that identifies the cause of the SSL initialization error.

JMS: Channel exits written in C and C++ called from JMS have changed behavior:

Channel exit programs, written in C or C++, now behave identically, whether they are called from WebSphere MQ classes for JMS or from a IBM WebSphere MQ MQI client. New Java channel exit interfaces offer improved function and performance.

You can now write channel exit classes to implement a new set of interfaces in the `com.ibm.mq.exits` package. Use the new interfaces to define classes to call as channel exits. The old Java channel exit interfaces, which are part of the IBM WebSphere MQ classes for Java , are still supported. The new interfaces have improved function and performance.

Related information:

Package com.ibm.mq.exits

Writing channel exits in Java for WebSphere MQ classes for JMS

Assigning channel exits for WebSphere MQ classes for JMS

JMS: Connection pooling:

The JMS connection pooling functionality has been removed from the IBM WebSphere MQ V7 classes for JMS.

The connection pooling functionality provided by the IBM WebSphere MQ V6 classes for JMS allowed connection handles (Hconns) to IBM WebSphere MQ to be stored in a pool where they were available for reuse. This functionality was implemented using the IBM WebSphere MQ classes for Java. This was possible because the IBM WebSphere MQ V6 classes for JMS were implemented as a layer of code on top of the IBM WebSphere MQ V6 classes for Java. In IBM WebSphere MQ V7.x, the implementation of IBM WebSphere MQ classes for JMS is no longer dependent on IBM WebSphere MQ classes for Java. This means that the IBM WebSphere MQ classes for JMS can no longer access features of the IBM WebSphere MQ classes for Java.

Java EE application servers provide JMS Connection Pools and JMS Session Pools for applications to use. When the application server was configured to use the IBM WebSphere MQ V6 classes for JMS, these JMS Connection and JMS Session Pools sat on top of the connection handle provided by the IBM WebSphere MQ classes for JMS. This led to some confusion about how the JMS Connection and Session Pools provided by application servers worked. Customers reported that there were still open connections from their application servers to IBM WebSphere MQ, even though the JMS Connection and JMS Session Pools were empty. The reason for this was because the connection handles associated with the JMS Connections and JMS Sessions, that had been removed from the application server's Connection and Session Pools, were now in the underlying IBM WebSphere MQ classes for JMS connection handle pool.

When the IBM WebSphere MQ V7 classes for JMS are used in Java EE Application Server environments, JMS Connection and IBM WebSphere MQ Session Pools are provided as follows:

- For Inbound Communications, the JMS Connection Pools and JMS Session Pools are implemented by the IBM WebSphere MQ JCA resource adapter.
- For Outbound Communications, the JMS Connection Pools and JMS Session Pools are implemented by the Java EE Application Server.

In all other environments, applications using the IBM WebSphere MQ V7 classes for JMS that require the use of JMS Connection and JMS Session pooling need to be modified to implement their own pooling logic.

JMS: Data conversion:

The default behavior from version 7.0.1.5 onwards, is for the client JVM to perform data conversion for a JMS client. This behavior is the same as if the client is connected to a version 6.0, or earlier release, of queue manager. For a version JMS client connected to a queue manager at version 7.0.0.0 to 7.0.1.4, the behavior can be different.

Connected to a version 6.0 queue manager, or earlier, data conversion is always performed by the client JVM. In version 7.0.0.0 to version 7.0.1.4 data conversion is either performed by the client JVM, or by the queue manager, but the JMS client has no direct control where conversion performed. From version 7.0.1.5 onwards, the JMS client controls where data conversion is performed. The default behavior for a client connected to a queue manager at version 7.0.1.5 onwards is the same as when the client is connected to a version 6.0 queue manager, or earlier.

As a significant improvement to the JMS client in WebSphere MQ Version 7.0, the JMS client can request that the queue manager performs data conversion. The change aligns the JMS client with all other WebSphere MQ clients in terms of carrying out data conversion in the queue manager. The queue manager uses operating system code page file sets to carry out data conversion, as it has done for many years. For details about data conversion and JMS clients, see JMS message conversion.

Between version 7.0.0.0 and 7.0.1.4 inclusive, queue manager data conversion is not optional. Conversion is performed on every JMS message sent to a client that contained JMS data with a defined message format, and which is not encoded in `1208` (UTF-8) or has numeric data not encoded in `Native` encoding. Both the client and the queue manager must be version 7.0, and the `PROVIDERVERSION` must be 7 for queue manager data conversion to be performed. If either is not version 7.0, `PROVIDERVERSION` is set to 6, and no queue manager data conversion is performed.

From version 7.0.1.5 onwards queue manager data conversion is optional. The default version 7.0.1.5 behavior is changed back to be the same as version 6.0, because queue manager conversion caused migration problems encountered in some cases. The problems that have been encountered, and their solutions are described in Table 75.

Table 75. Queue manager data conversion problems and solutions

<table>
<tr><th rowspan="2">Problem</th><th colspan="3">Solution</th></tr>
<tr><th>7.0.0.0 → 7.0.1.3</th><th>7.0.1.3 with IZ67359</th><th>7.0.1.4 with IC72897 or 7.0.1.5 onwards</th></tr>
<tr><td>z/OS new line differences
(Differences in mapping new lines between code pages used by Java and code pages used by WebSphere MQ)</td><td>No work-around</td><td>No work-around</td><td>You have 2 options:
1. <b>The default behavior resolves the problem</b>
2. Set RECEIVECONVERSION to WMQ_RECEIVE_CONVERSION_CLIENT_MSG</td></tr>
<tr><td rowspan="2">Queue manager and application both convert formatted message, and the application assumes the CCSID of the header.</td><td colspan="3">Alter the application to check the CCSID of the message before performing conversion</td></tr>
<tr><td></td><td>Set ReceiveCCSID system property to the value the application expects.</td><td>You have 2 options:
1. <b>The default behavior resolves the problem</b>
2. Set RECEIVECONVERSION to WMQ_RECEIVE_CONVERSION_QMGR and RECEIVECCSID to the value the application expects.</td></tr>
<tr><td rowspan="2">Queue manager does not recognize the name in the message data format property.</td><td colspan="3">Implement a null conversion exit</td></tr>
<tr><td></td><td></td><td>You have 2 options:
1. <b>The default behavior resolves the problem</b>
2. Set RECEIVECONVERSION to WMQ_RECEIVE_CONVERSION_CLIENT_MSG</td></tr>
<tr><td>Queue manager unable to convert to 1208.</td><td>No work-around</td><td>Set ReceiveCCSID system property to a value the queue manager can convert to.</td><td>You have 2 options:
1. <b>The default behavior resolves the problem</b>
2. Set RECEIVECONVERSION to WMQ_RECEIVE_CONVERSION_QMGR and RECEIVECCSID to a value the queue manager can convert to.</td></tr>
</table>

Note:

1. The default behavior in version 7.0.1.5 (or version 7.0.1.4 with IC72897) has reverted to the behavior in version 6.0. It is possible that you have written a version 7.0 JMS client that takes advantage of

queue manager data conversion. For the client to continue to work correctly, you must set the property RECEIVECONVERSION to WMQ_RECEIVE_CONVERSION_QMGR. See JMS message conversion for examples.

2. The MQ link receiver between a WebSphere MQ channel and the WebSphere Application Server service integration bus does not use the WebSphere MQ classes for JMS as its JMS provider. It does not call a data conversion exit. It is unaffected by these migration considerations.

Related information:

JMS message conversion

Properties of WebSphere MQ classes for JMS objects

The WebSphere MQ classes for JMS configuration file

When to use PROVIDERVERSION
In two cases you must override the default selection of **PROVIDERVERSION** for the WebSphere MQ classes for JMS to work correctly.

JMS: Dots in property names:

JMS property names must not contain dots. The rules for naming JMS properties in the JMS specification follow the rules for naming Java identifiers. One of the rules is that identifiers must not contain dots. IBM WebSphere MQ Version 6.0 documented the rule, but did not enforce it. Releases following Version 6.0 enforce the rule.

A JMS program calling Version 6.0 of IBM WebSphere MQ classes for JMS can create JMS property names containing dots. The same program, calling a version of IBM WebSphere MQ classes for JMS later than Version 6.0 receives an exception. A DetailedMessageFormatException is thrown by the Set*type*Property methods of JMSMessage.

The same application calling Version 6.0 of IBM WebSphere MQ classes for JMS does not throw an exception, even if connected to a queue manager running Version 7.5 .

JMS: Error code and exception message changes:

Compared to Version 6.0 of IBM WebSphere MQ classes for JMS, most error codes, and exception messages, have changed. The reason for these changes is that IBM WebSphere MQ classes for JMS now has a layered architecture. Exceptions are thrown from different layers in the code.

You must modify your application if it parses or tests exception messages returned by the Throwable.getMessage method, or error codes returned by the JMSException.getErrorCode method.

Examples of changes

For example, an application tries to connect to a queue manager that does not exist. Version 6.0 of WebSphere MQ classes for JMS threw a JMSException exception with the following information.

```
MQJMS2005: Failed to create MQQueueManager for 'localhost:QM_test'.
```

This exception contained a linked MQException exception with the following information.

```
MQJE001: Completion Code 2, Reason 2058
```

WebSphere MQ classes for JMS now throws a JMSException exception with the following information.

```
Message : JMSWMQ0018: Failed to connect to queue manager 'QM_test' with
          connection mode 'Client' and host name 'localhost'.
Class : class com.ibm.msg.client.jms.DetailedJMSException
Error Code : JMSWMQ0018
```

```
Explanation : null
User Action : Check the queue manager is started and if running in client mode,
              check there is a listener running. Please see the linked exception
              for more information.
```

This exception contains a linked MQException exception with the following information.

```
Message : JMSCMQ0001: WebSphere MQ call failed with compcode '2' ('MQCC_FAILED')
          reason '2058' ('MQRC_Q_MGR_NAME_ERROR').
Class : class com.ibm.mq.MQException
Completion Code : 2
Reason Code : 2058
```

JMS: Exception listeners changes in Version 7.0:

In later releases of the product, JMS exception listeners behave differently than they did in Version 6.0. Applications might receive more or fewer exceptions than they did in Version 6.0.

In Version 6.0 of IBM WebSphere MQ, an exception listener was called to inform the application of any error condition that occurred asynchronously to the application execution. During the processing of a message for an asynchronous consumer, the application might have had no other means to discover the exception. The errors that resulted in the exception listener being called included connection broken events, or if an attempt was made to process an unreadable message.

Previously, with Version 6.0 JMS clients, a property had to be explicitly set if connection broken type exceptions that occurred during synchronous API calls were to be passed to the application's ExceptionListener code. However, from Version 7.0, connection broken exceptions are expected to be passed, without the need to set any properties, to an application's ExceptionListener regardless of whether they were making synchronous API calls or receiving messages asynchronously to a MessageListener.

From Version 7.0, you can set the **ASYNC_EXCEPTIONS** property on JMS Connection Factory objects using the method:

```
setIntProperty(JmsConstants.ASYNC_EXCEPTIONS, int)
```

.

V7.5.0.8 Before APAR IT14820, included from IBM WebSphere MQ Version 7.5.0, Fix Pack 8, the default value of this property was ASYNC_EXCEPTIONS_ALL. With APAR IT14820, the default value is changed to ASYNC_EXCEPTIONS_CONNECTIONBROKEN. For more information, see “JMS: Exception listener changes in Version 7.5” on page 560.

If you set the **ASYNC_EXCEPTIONS** property to ASYNC_EXCEPTIONS_ALL, the exception listener is called for all broken connection exceptions. The exception listener is also called for all exceptions that occur outside the scope of a synchronous JMS API call. Creating a JMS Connection with an ExceptionListener from a JMS Connection Factory that has **ASYNC_EXCEPTIONS** set to ASYNC_EXCEPTIONS_ALL should provide broadly the same behavior as a Version 6.0 JMS client that had the activateExceptionListener Java system property set to the value true.

If you set the **ASYNC_EXCEPTIONS** property to ASYNC_EXCEPTIONS_CONNECTIONBROKEN, only exceptions indicating a broken connection are sent to the exception listener. These exceptions include connection broken exceptions occurring both synchronously and asynchronously. They do not include any other asynchronous errors such as for unreadable messages. In this mode, if the exception listener is triggered, the connection can be considered to have failed. It is no longer possible to use the connection to send or receive messages.

Applications must take appropriate action, such as attempting to remake the connection, when exception listener calls are made.

Note: V7.5.0.8 From IBM WebSphere MQ Version 7.5.0, Fix Pack 8, further changes are made for JMS exception listeners. For more information, see “JMS: Exception listener changes in Version 7.5” on page 560.

Related information:

Exceptions in WebSphere MQ classes for JMS

ASYNCEXCEPTION

JMS: Integration of IBM WebSphere MQ classes for JMS with WebSphere Application Server:

You can configure IBM WebSphere MQ as a JMS provider for WebSphere Application Server. The way the IBM WebSphere MQ messaging provider is configured has changed between WebSphere Application Server V6.0/6.1 and V7.0. This topic describes the changes, and different configurations you can create to run different versions of WebSphere Application Server and IBM WebSphere MQ.

The way IBM WebSphere MQ is integrated with WebSphere Application Server has changed between WebSphere Application Server V6.0/6.1 and V7.0.

WebSphere Application Server V7.0

IBM WebSphere MQ classes for JMS are built in to WebSphere Application Server V7.0 as a JCA 1.5 resource adapter, `wmq.jmsra.rar`. To use the IBM WebSphere MQ resource adapter as the messaging provider in WebSphere Application Server, configure activation specifications and connection factories to use the IBM WebSphere MQ messaging provider.

`wmq.jmsra.rar` is also shipped with IBM WebSphere MQ. It is identical to the `wmq.jmsra.rar` file shipped with WebSphere Application Server. `wmq.jmsra.rar` is shipped with IBM WebSphere MQ to install as a JCA 1.5 resource adapter on application servers from different suppliers.

Note: The WebSphere Application Server instructions for installing JCA resource adapters from other suppliers do not apply to the IBM WebSphere MQ resource adapter. Should you mistakenly install `wmq.jmsra.rar` on WebSphere Application Server as a JCA 1.5 resource adapter, it results in uncertainty about which version of the adapter is used.

If you do not install IBM WebSphere MQ on the same server as WebSphere Application Server, you can use the IBM WebSphere MQ classes for JMS in client mode only. If you install IBM WebSphere MQ on the same server as WebSphere Application Server, you have the choice of using the IBM WebSphere MQ classes for JMS in either client mode or bindings mode. Use the IBM WebSphere MQ messaging provider general property **`Native Library Path`**, to locate the native IBM WebSphere MQ Java libraries installed on the same server as the application server.

In bindings mode, the IBM WebSphere MQ queue manager must be at least at fix pack 6.0.2.5 (6.0.2.7 on z/OS). In client mode the IBM WebSphere MQ classes for Java can connect to any queue manager at V6.0 or higher.

For full IBM WebSphere MQ Version 7.5 functionality, use WebSphere Application Server V7.0 with IBM WebSphere MQ Version 7.5. If the application server connects to a V6.0 queue manager, either locally or remotely, the IBM WebSphere MQ classes for JMS run in migration mode; see “Mixed configurations” on page 619.

The IBM WebSphere MQ JCA resource adapter, which is shipped with WebSphere Application Server, is maintained by applying WebSphere Application Server fix packs; see Which version of WebSphere MQ is shipped with WebSphere Application Server. Maintain the IBM WebSphere MQ server libraries in the normal way, by applying IBM WebSphere MQ fix packs, or PTFs on z/OS.

If you migrate profiles from WebSphere Application Server 7.0.0.0 to later maintenance levels, or from later maintenance levels to the 7.0.0.0 level, you must manually adjust the resource adapter configuration; see Maintaining the WebSphere MQ JCA resource adapter.

WebSphere Application Server V6.0/V6.1

The IBM WebSphere MQ classes for JMS, com.ibm.mq.jar, com.ibm.mqjms.jar, and dhbcore.jar, are installed by WebSphere Application Server into *$WAS_INSTALL_ROOT*\lib\WMQ\java\lib. Using these classes, JMS applications connect to IBM WebSphere MQ in client mode.

If you do not install IBM WebSphere MQ on the same server as WebSphere Application Server, you can use the IBM WebSphere MQ classes for JMS in client mode only. If you install IBM WebSphere MQ on the same server as WebSphere Application Server, you have the choice of using the IBM WebSphere MQ classes for JMS in either client mode or bindings mode.

To use the IBM WebSphere MQ classes for JMS installed by IBM WebSphere MQ, set the WebSphere Application Server environment variable *$MQ_INSTALL_ROOT* to point to the IBM WebSphere MQ installation. You can then use the libraries installed by IBM WebSphere MQ to connect to IBM WebSphere MQ in either client or bindings mode.

You can use IBM WebSphere MQ Version 7.5 with WebSphere Application Server V6.0 or V6.1. The default configuration, which uses the IBM WebSphere MQ classes for JMS installed by WebSphere Application Server, is restricted by the client to IBM WebSphere MQ Version 6.0 functionality. If you install either the IBM WebSphere MQ Version 7.5 client or server on the application server, you can access IBM WebSphere MQ Version 7.5 functionality. With the combination of IBM WebSphere MQ Version 7.5 and WebSphere Application Server V6.0/V6/.1, you can use the integrated publish/subscribe API, shared channels, queue manager selectors, and other V7.0 enhancements; see "Mixed configurations."

wmq.jmsra.rar is also shipped with IBM WebSphere MQ. It is identical to the wmq.jmsra.rar file shipped with WebSphere Application Server. wmq.jmsra.rar is shipped with IBM WebSphere MQ to install as a JCA 1.5 resource adapter on application servers from different suppliers. You must not configure a JCA resource adapter on WebSphere Application Server V6.0 or V6.1 using the IBM WebSphere MQ resource adapter, wmq.jmsra.rar. If you do so, and then try to start a JMS application, it results in Java errors.

Maintain the IBM WebSphere MQ classes for JMS shipped with WebSphere Application Server by applying WebSphere Application Server fix packs; see Which version of WebSphere MQ is shipped with WebSphere Application Server. If you are using the libraries installed by a IBM WebSphere MQ installation by setting *$MQ_INSTALL_ROOT*, maintain the IBM WebSphere MQ libraries in the normal way, by applying IBM WebSphere MQ fix packs, or PTFs on z/OS.

Mixed configurations

When you upgrade WebSphere Application Server from V6.0/6.1 to V7.0 or IBM WebSphere MQ from Version 6.0 to Version 7.5, you can run with different versions of WebSphere Application Server and IBM WebSphere MQ, as well as upgrading both. You must take note of any functional and performance implications of running with a mixed configuration. Mixed configurations are supported to cater for complex migration scenarios.

WebSphere Application Server V7.0 and IBM WebSphere MQ Version 6.0

WebSphere Application Server V7.0 always uses the Version 7.0 IBM WebSphere MQ classes for JMS installed with the application server as the IBM WebSphere MQ message provider. The IBM WebSphere MQ message provider connects to a local or remote queue manager running IBM WebSphere MQ Version 6.0. The Version 7.0 capabilities in the IBM WebSphere MQ resource adapter are not available when connected to the Version 6.0 queue manager. As a result, the IBM WebSphere MQ classes for JMS run in migration mode, which has a performance and function cost. Migration mode is set automatically. You can configure it manually by setting the IBM WebSphere MQ classes for JMS property PROVIDERMODE=6.

You must apply a fix to the IBM WebSphere MQ resource adapter to run IBM WebSphere MQ classes for JMS in this configuration. The minimum WebSphere Application Server fix pack required is 7.0.0.9. The IBM WebSphere MQ resource adapter is provided by WebSphere Application Server, so you must obtain the fix pack from WebSphere Application Server rather than IBM WebSphere MQ. If you are using a different application server, you must apply IBM WebSphere MQ fix pack 7.0.1.1 to update `wmq.jmsra.rar` to the same level.

If you had installed IBM WebSphere MQ Version 6.0 on the same server as WebSphere Application Server V6.0, you can continue to use both client and bindings mode.

With WebSphere Application Server V6.0/V6.1 and IBM WebSphere MQ Version 6.0 installed, you would have set the WebSphere Application Server environment variable, *`$MQ_INSTALL_ROOT`*, to the IBM WebSphere MQ Version 6.0 installation directory. WebSphere Application Server V6.0/V6.1 loads the IBM WebSphere MQ classes for JMS from *`$MQ_INSTALL_ROOT`* to resolve references to JMS classes. WebSphere Application Server V7.0 does not use *`$MQ_INSTALL_ROOT`*. WebSphere Application Server V7.0 loads the IBM WebSphere MQ classes for JMS from the IBM WebSphere MQ resource adapter shipped with WebSphere Application Server V7.0.

Some behavior of the IBM WebSphere MQ classes for JMS has changed between IBM WebSphere MQ Version 6.0 and IBM WebSphere MQ Version 7.0. Even if the JMS application is connected to a Version 6.0 queue manager, because WebSphere Application Server V7.0 uses the Version 7.0 IBM WebSphere MQ classes for JMS, there are some differences in JMS client behavior.

- The MQException class is packaged in `com.ibm.wmq.jmqi.jar`, not `com.ibm.mq.jar`.
  - Applications compiled with Java 1.5, or greater, automatically resolve class references to `com.ibm.mq.jmqi.jar`.
- JMS exceptions and errors are handled differently; see "JMS: Error code and exception message changes" on page 616.

WebSphere Application Server V6.0/6.1 and IBM WebSphere MQ Version 7.5

To use WebSphere Application Server V6.0/6.1 with the Version 7.5 IBM WebSphere MQ classes for JMS, you must install either the IBM WebSphere MQ Version 7.5 client, or Version 7.5 server, on the same server as WebSphere Application Server. Set *`$MQ_INSTALL_ROOT`* to the IBM WebSphere MQ installation directory path.

You might encounter JMS application migration problems, as a result of changes to the IBM WebSphere MQ classes for JMS. If your solution requires connection to a Version 7.5 queue manager, you might be able to bypass some problems by continuing to use the Version 6.0 IBM WebSphere MQ classes for JMS. Alternatively, install the Version 6.0 WebSphere MQ JMS client on the same server as the application server and set *`$MQ_INSTALL_ROOT`* to the WebSphere MQ installation directory. Connect the JMS application to the Version 7.5 queue manager in client mode.

Note: You cannot connect the Version 6.0 IBM WebSphere MQ classes for JMS, installed with WebSphere Application Server, to IBM WebSphere MQ Version 7.5 installed on the same server, in bindings mode.

The WebSphere Application Server V6.0/V6.1 administration console only configures IBM WebSphere MQ Version 6.0 capabilities. To configure new IBM WebSphere MQ Version 7.5 properties, you must use custom properties.

Miscellaneous changes between Version 6.0 to Version 7.5 affecting JMS

In this section changes that affect JMS applications using the WebSphere MQ messaging provider managed by WebSphere Application Server are described. Changes to the WebSphere MQ implementation of JMS in Version 7.5 that affect the migration of JMS applications from IBM WebSphere MQ Version 6.0 to IBM WebSphere MQ Version 7.5, are described in other topics.

The Target client field is no longer displayed in the WebSphere Application Server Administration Console.

The **Target client** field on a JMS destination in V6.0/V6.1 specified whether the message recipient expected the body of a JMS message was a JMS message or a WebSphere MQ message. Examples of JMS messages are JMS bytes, JMS text, and JMS object.

In WebSphere Application Server V7.0, the same function is controlled by the check box **Append RFH version 2 headers to messages sent to this destination**. See "Target client" is field not displayed in the WebSphere Application Server Administration Console V7.

If you are sending messages to a WebSphere MQ application, that does not expect JMS messages, clear the check box. If you select the check box, messages are constructed with an `RFH2` header specifying the JMS message type.

A number of JMS issues are resolved, including use of multi-instance queue managers.

A number of JMS issues are resolved in later WebSphere Application Server fix packs.

Problems using multi-instance queue managers, are fixed in WebSphere Application Server fix pack 7.0.0.13 (WebSphere MQ fix pack 7.0.1.3). Set the new custom properties, `XMSC_WMQ_CONNECTION_NAME_LIST` for connection factories, and `connectionNameList` for activation specifications, to create a comma-separated lists of `host(port)` names to connect to a multi-instance queue manager.

Automatic client reconnection is not supported by WebSphere Application Server.

Asynchronous consume, browse with mark, and Application Server Facilities

Message Driven Beans (MDBs) using Application Server Facilities (ASF) use asynchronous consume rather than polling to read messages asynchronously. Asynchronous consume is a IBM WebSphere MQ Version 7.0 enhancement.

The destination is now able to browsed with mark on non z/OS platforms, eliminating problems of message contention, and lower priority messages being marooned by constantly arriving higher priority messages.

As a result ASF is more efficient on non z/OS platforms than it was with V6.0. (WebSphere MQ for z/OS efficiently received messages asynchronously in V6.0.)

Activation specifications

Use activation specifications with MDBs, rather than message listeners, to consume messages in WebSphere Application Server V7.0. Activation specifications are better than message listeners:

- Activation specifications are part of the JCA 1.5 specification standard.
- Using activation specification eliminates configuring a connection factory for inbound messages.
- Activation specifications can be defined at node or cell scope, not just server scope. If you have multiple servers, you require only one activation specification, but multiple message listener definitions.

Message listener ports are still supported. The WebSphere Application Server administration console has a wizard, and a command, to migrate a message listener to an activation specification.

Terminology

WebSphere Application Server uses the term “WebSphere MQ messaging provider”. The WebSphere MQ messaging provider is the same code as the IBM WebSphere MQ classes for JMS. WebSphere MQ also provides the IBM WebSphere MQ classes for JMS as JCA 1.5 resource adapter for installing on other application servers. All these terms refer to the same code.

Related information:

Which version of WebSphere MQ is shipped with WebSphere Application Server?

IC64098: APPLICATION DOES NOT AUTOMATICALLY RECONNECT TO THE QUEUE MANAGER IF CONNECTION IS LOST WHEN USING THE MQ RESOURCE ADAPTER

Fix Central

PK87026: CUMULATIVE MAINTENANCE FIXPACK 6.0.2.7 FOR THE JMS FEATURE OF WEBSPHERE MQ FOR Z/OS VERSION 6.

Maintaining the WebSphere MQ resource adapter

When to use ASF and non-ASF modes to process messages in WebSphere Application Server

WebSphere MQ classes for JMS Application Server Facilities

Using the WebSphere MQ messaging provider in WebSphere Application Server V7: Part 1: Introducing the new WebSphere MQ messaging provider

JMS: Messaging provider:

The IBM WebSphere MQ classes for JMS normally call the integrated publish/subscribe interface by default. In Version 6.0 the classes used the queued command message interface.

With the new **PROVIDERVERSION** JMS property an application can select whether the IBM WebSphere MQ classes for JMS call the the integrated or the queued command message publish/subscribe interface; see When to use PROVIDERVERSION. By default the integrated publish/subscribe interface is normally called, but depending on the context, the IBM WebSphere MQ classes for JMS might call the queued interface; see Rules for selecting the WebSphere MQ messaging provider mode.

Related reference:

"JMS **PROVIDERVERSION** property" on page 722

The JMS **PROVIDERVERSION** property selects whether a Java application publishes and subscribes using the queued command message interface, or the integrated call interface.

JMS: Migration of subscriptions to named streams:

In Version 6.0 publish/subscribe, you can create subscriptions to topics in named streams, as well as to topics in the default, or unnamed, stream. Version 7.5 publish/subscribe does not have named streams. It emulates streams by modifying the topic string of a subscription that is associated with a stream. The result is, for most applications, the migration of subscription names from Version 6.0 to Version 7.5 is automatic. In the following cases, make changes to the results of the automatic migration.

- Adding streams, after completing migration.
- Migrating streams that have names that clash with topic names .
- Adding of modifying properties of the subscription, such as access control or clustering.

The relationship between queued publish/subscribe streams and integrated publish/subscribe topics

In queued publish/subscribe, publications can be divided into different streams. Publications on same topic, published to different streams, are separate from one another. A subscriber on one stream cannot subscribe to publications to another stream. Each stream is implemented as a separate stream queue. There is a common, unnamed, stream queue that is called `SYSTEM.BROKER.DEFAULT.STREAM`.

In integrated publish/subscribe, publications cannot be separated into different streams. Two topics, to be distinct, must have two different topic strings.

To map between the two topic models, the stream name is prefixed to the topic string used by integrated publish/subscribe. Suppose a Version 6.0 publisher creates a publication on stream `UKSPORTS` with the topic string `football`. Generally, to a Version 7.5 subscriber, the publication has the topic string `UKSPORTS/football`. For a JMS client program, the same `BrokerPubQ` connection factory property that named the stream `UKSPORTS` in Version 6.0, provides the root topic name `UKSPORTS` in Version 7.5. No client program change is required.

In practice, mapping between streams and topics might not be as simple as the stream name being the same as the root topic name. To cater for the possible collision of stream names with existing root topic strings, the stream name corresponds to a topic object name. The steam name maps to the topic string defined for the topic object. In the preceding example, the Version 7.5 queue manager must have a topic object called `UKSPORTS` with a topic string `UKSPORTS`. If the topic string was `GBSPORTS`, the Version 6.0 publication to `football`, on the `UKSPORTS` stream, would be mapped to the Version 7.5 topic `GBSPORTS/football`.

During the migration of a Version 6.0 queue manager to Version 7.5, if you have used Version 6.0 publish/subscribe, you must run the command **strmqbrk**. **strmqbrk** migrates the Version 6.0 publish/subscribe resources that are defined on the queue manager. It automatically configures stream to topic mapping on Version 7.5 for any streams you have defined on Version 6.0. For example, if the publish/subscribe broker had a stream, UKSPORTS, **strmqbrk** creates a topic object `UKSPORTS` with a topic string `UKSPORTS`. It also modifies the namelist, `SYSTEM.QPUBSUB.QUEUE.NAMELIST`; see Figure 99 on page 624

How JMS clients refer to publications to named streams in Version 7.5

In Version 6.0, a JMS client set `BrokerPubQ` to the stream name to use a named stream. In Version 7.5, `BrokerPubQ` is interpreted by the queue manager as the name of a WebSphere MQ administrative topic object.

The queue manager uses the topic object name defined by `BrokerPubQ` to set the topic name property of a JMS topic. The queue manager finds the administrative topic object named in `BrokerPubQ` and extracts its topic string. It prefixes the topic name passed to the JMS topic with the topic string.

For example, suppose you migrated the publish/subscribe stream `UKSPORTS` from Version 6.0, and subscribed to a JMS topic, with the JMS topic string `football`. In Version 7.5, the queue manager adds `UKSPORTS` to the topic name passed to the JMS topic. In Version 7.5, the JMS application still sets the topic name to `football`, but the queue manager sets the JMS topic name to `UKSPORTS/football`.

The JMS client requires no code change. If the automatic migration performed by **strmqbrk** was successful, and suitable for your installation, then no manual administrative changes are required either.

You might make further administrative changes:

- You must add streams manually if you introduce new streams after migrating to Version 7.5.
- If you are migrating streams that have names which clash with topic names in the Version 7.5 topic space.

  You can redefine the topic object created by **strmqbrk**.

  Delete the topic object that was created, and create a topic object with the same topic name, but a different topic string.
- Use the root topic object, `UKSPORTS`, to configure additional properties of the topic, such as access control, or clustering.

Tip: You can set the connection factory property `BrokerPubQ` in Version 7.5 to restrict a JMS client to a specific topic space. The topic space is restricted to the topic tree rooted in the topic name defined in the topic object named in `BrokerPubQ`. The topic space can be set administratively, and changed, without changing the JMS client.

The migration command **strmqbrk** adds existing streams to a Version 7.5 queue manager automatically. You can create a stream to use with Version 7.5 queued publish/subscribe, after you have migrated the publish/subscribe resources. Follow the example, Figure 99.

```
define qlocal('UKSPORT')
     1 : define qlocal('UKSPORT')
AMQ8006: WebSphere MQ queue created.
define topic('UKSPORT') topicstr('UKSPORT') wildcard(BLOCK)
     2 : define topic('UKSPORT') topicstr('UKSPORT') wildcard(BLOCK)
AMQ8690: WebSphere MQ topic created.
alter namelist(SYSTEM.QPUBSUB.QUEUE.NAMELIST)
      NAMES('UKSPORT', 'SYSTEM.BROKER.DEFAULT.STREAM', 'SYSTEM.BROKER.ADMIN.STREAM')
     3 : alter namelist(SYSTEM.QPUBSUB.QUEUE.NAMELIST)
         NAMES('UKSPORT', 'SYSTEM.BROKER.DEFAULT.STREAM', 'SYSTEM.BROKER.ADMIN.STREAM')
AMQ8551: WebSphere MQ namelist changed.
```

Figure 99. Adding a stream to a Version 7.5 queue manager

Related reference:

"Publish/Subscribe: Subscription names" on page 656
In IBM WebSphere MQ Version 6.0, a subscription name is unique only within the stream and not across the queue manager. A subscription name must now be unique across the queue manager.

"Publish/Subscribe: Streams" on page 655
Streams are used in Version 6.0 to isolate publish/subscribe applications. Streams are now emulated. Migration from Version 6.0 streams is automatic. Existing Version 6.0 applications that use streams continue to work unchanged when migrated to, or interoperating with, later versions of IBM WebSphere MQ. In some unusual cases, you might need to modify the results of the automatic migration. If you introduce new streams to Version 6.0 applications, you must configure the emulated streams manually.

Related information:

Adding a stream, version 7.0.1
You can add streams manually so that they coexist with streams migrated from Version 6.0 queue managers.

Deleting a stream, version 6.0

Stream and topics, version 7.0.1
Queued publish/subscribe has the concept of a publication stream that does not exist in the integrated publish/subscribe model. In queued publish/subscribe, streams provide a way of separating the flow of information for different topics. In IBM WebSphere MQ Version 6.0, a stream is implemented as a queue, defined at each broker that supports the stream. Each queue has the same name (the name of the stream). From IBM WebSphere MQ Version 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

JMS: Receiving `MQRFH2` headers in messages sent from JMS applications:

In Version 6.0, if a JMS message is received by a non-JMS application, the JMS properties are returned in an `MQRFH2` header. In later versions of IBM WebSphere MQ, the JMS properties might be returned either as message properties, or as an `MQRFH2` header.

To preserve the IBM WebSphere MQ Version 6.0 behavior of returning JMS message properties in an `MQRFH2` header you have two choices.

1. Leave the **`MQGMO`** property option as its default value, `MQGMO_PROPERTIES_AS_Q_DEF`, and leave the queue **`PROPCTL`** attribute set to its default value, `COMPAT`.
2. Set the **`MQGMO`** property option to `MQGMO_PROPERTIES_COMPATIBILITY`.

Related reference:

"**PROPCTL** queue options" on page 731
Use the **PROPCTL** queue attribute to control how message properties are returned to an application that calls MQGET without setting any MQGMO message property options.

Related information:

MQRFH2 - Rules and formatting header 2

JMS: Tracing and error reporting:

JMS tracing and error reporting has been extended. For compatibility with Version 6.0 the Java property MQJMS_TRACE_LEVEL is retained, but the trace differs.

IBM WebSphere MQ classes for JMS now contains a class that an application can use to control tracing. An application can start and stop tracing, specify the required level of detail in a trace, and customize trace in various ways. For example, you can now configure trace by using properties that you specify in a client configuration file. For more information, see Tracing IBM WebSphere MQ classes for JMS applications.

In version 6.0, the Java property **MQJMS_TRACE_LEVEL** turned on JMS trace. It has three values:

on
Traces IBM WebSphere MQ classes for JMS calls only.

base
Traces both IBM WebSphere MQ classes for JMS calls and the underlying IBM WebSphere MQ classes for Java calls.

off
Disables tracing.

Setting **MQJMS_TRACE_LEVEL** to on or base produces the same results as setting the **com.ibm.msg.client.commonservices.trace.status** property to on.

Setting the property, **MQJMS_TRACE_DIR** to somepath/tracedir is equivalent to setting the **com.ibm.msg.client.commonservices.trace.outputName** property to somepath/tracedir/mqjms_%PID%.trc.

JMS: ResourceAdapter object configuration:

When a WebSphere Application Server connects to IBM WebSphere MQ it creates Message Driven Beans (MDBs) using JMS connections. These MDBs can no longer share one JMS connection. The configuration of ResourceAdapter object is migrated so that there is a single MDB for each JMS connection.

Changed ResourceAdapter properties

connectionConcurrency
The maximum number of MDBs to share a JMS connection. Sharing connections is not possible and this property always has the value 1. Its previous default value was 5.

maxConnections
This property is the number of JMS connections that the resource adapter can manage. In Version 7.5, it also determines the number of MDBs that can connect because each MDB requires one JMS connection. The default value of maxConnections is now 50. Its previous default value was 10.

If connectionConcurrency is set to a value greater than 1, the maximum number of connections supported by the resource adapter is scaled by the value of connectionConcurrency. For example, if maxConnections is set to 2 and connectionConcurrency is set to 4, the maximum number of connections supported by the resource adapter is 8. As a result, connectionConcurrency is set to 1 and maxConnections is set to 8.

If connectionConcurrency is set to a value greater than 1, it is adjusted automatically. To avoid automatic adjustment, set connectionConcurrency to 1. You can then set maxConnections to the value you want.

The scaling mechanism ensures that sufficient connections are available for existing deployments whether you have changed them in your deployment, configuration, or programs.

If the adjusted `maxConnections` value exceeds the `MAXINST` or `MAXINSTC` attributes of any used channel, previously working deployments might fail.

The default value of both channel attributes equates to unlimited. If you changed them from the default value, you must ensure that the new `maxConnections` value does not exceed `MAXINST` or `MAXINSTC`.

Related information:

Configuration of the ResourceAdapter object

Maximum instances (MAXINST)

Maximum instances per client (MAXINSTC)

JMS: JMS_IBM_Character_Set:

Prior to IBM WebSphere MQ V7.5, applications using IBM WebSphere MQ messaging provider migration mode could set the JMS_IBM_Character_Set property of a message to a numerical Coded Character Set Identifier.

When the message was sent, the Coded Character Set Identifier stored in the JMS_IBM_Character_Set property was mapped to the MQMD field CodedCharacterSetID.

When using the IBM WebSphere MQ V7.5 classes for JMS, a JMSException containing the message:

```
MQJMS1006: invalid value for 'JMS_IBM_Character_Set': '<number>'
```

is thrown if an application tries to send a message that has the JMS_IBM_Character_Set property set to a numerical Coded Character Set Identifier.

The JMS_IBM_Character_Set property must be set to the Java character set string that maps to the Coded Character Set Identifier that the application wants to use. For more information, see Mapping JMS fields onto WebSphere MQ fields (outgoing messages)

Related information:

Mapping JMS fields onto WebSphere MQ fields (outgoing messages)

MQI reason code changes

The reason codes that are returned for a few specific errors have changed. If an application checks any of these reason codes to take a specific action, you must check whether the application must be modified.

MQCONN and MQCONNX

Some more specific reason codes for failed MQCONN and MQCONNX calls are now returned by IBM WebSphere MQ.

If an MQCONN or MQCONNX call failed to connect to a queue manager in IBM WebSphere MQ Version 6.0, it issued reason code 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE. It can still issue `MQRC_Q_MGR_NOT_AVAILABLE`, or one of the following more specific reason codes.

- 2537 (09E9) (RC2537): MQRC_CHANNEL_NOT_AVAILABLE
- 2538 (09EA) (RC2538): MQRC_HOST_NOT_AVAILABLE
- 2539 (09EB) (RC2539): MQRC_CHANNEL_CONFIG_ERROR
- 2540 (09EC) (RC2540): MQRC_UNKNOWN_CHANNEL_NAME

If an application takes specific actions based on the reason code, change the application to take account of the additional reason codes.

An application that connects to multiple queue managers in the same process and uses MQCNO_FASTPATH_BINDING might fail with an error and reason code MQRC_FASTPATH_NOT_AVAILABLE; see "Connect to multiple queue managers and use MQCNO_FASTPATH_BINDING" on page 573.

MQOPEN and MQPUT1

The product documentation in previous versions of IBM WebSphere MQ warned about defining the default transmission queue as SYSTEM.CLUSTER.TRANSMIT.QUEUE. In Version 7.5, an attempt to open the default transmission queue, defined as SYSTEM.CLUSTER.TRANSMIT.QUEUE, results in the error MQRC_DEF_XMIT_Q_USAGE_ERROR.

In Version 7.5, an attempt to alter the queue manager attribute **DEFXMITQ** to SYSTEM.CLUSTER.TRANSMIT.QUEUE results in an error. The PCF reason code is 3269 (0CC5) (RC3269): MQRCCF_DEF_XMIT_Q_CLUS_ERROR.

MQSUB, MQPUT, MQPUT1

As a result of changes to message selector processing, applications that check explicitly for reason code 2459 (099B) (RC2459): MQRC_SELECTOR_SYNTAX_ERROR must check for, and handle, the reason code 2551 (09F7) (RC2551): MQRC_SELECTION_NOT_AVAILABLE.

MQPUT1

If a IBM WebSphere MQ MQI client application issues MQPUT1 in sync point, and later commits the message, some failures are now returned in the MQCMIT call. In Version 6.0 the failures are returned by MQPUT1. See the scenario in "Example call sequence that demonstrates the change in behavior" on page 603 in "MQI client: MQPUT1 sync point behavior change" on page 602.

Related reference:

"MQI client: MQPUT1 sync point behavior change" on page 602
An MQPUT1 call by a IBM WebSphere MQ MQI client application that failed in IBM WebSphere MQ Version 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

Related information:

2537 (09E9) (RC2537): MQRC_CHANNEL_NOT_AVAILABLE

2538 (09EA) (RC2538): MQRC_HOST_NOT_AVAILABLE

2539 (09EB) (RC2539): MQRC_CHANNEL_CONFIG_ERROR

2540 (09EC) (RC2540): MQRC_UNKNOWN_CHANNEL_NAME

2551 (09F7) (RC2551): MQRC_SELECTION_NOT_AVAILABLE

2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

3269 (0CC5) (RC3269): MQRCCF_DEF_XMIT_Q_CLUS_ERROR

MQRFH2 migration, message properties, and property folders

The addition of message properties to the MQI in IBM WebSphere MQ Version 7.0 can result in some applications that processed MQRFH2 folders satisfactorily in Version 6.0, getting different results in Version 7.5. If you leave the new queue and channel **PROPCTL** attribute at its default setting of COMPAT, most applications that process MQRFH2 folders are unaffected.

Message properties are transferred either in the message descriptor, MQMD, or if there is an MQRFH2 header, in *NameValueData*. Some message properties are only transferred in *NameValueData*.

If a message property is transferred in *NameValueData*, it is transferred in special type of folder, called a property folder. Any properties can be accessed using the MQINQMP and MQSETMP MQI calls.

A folder contains either name/value pairs, and is known as an ordinary folder, or it contains message properties and it is a properties folder. The syntax of *NameValueData* makes clear which folders are ordinary folders, and which are property folders; see NameValueData (MQCHARn). In short, a property folder has two forms. It is either a folder with the attribute `contents='properties'`, or it is one of a special set of folders that are designated property folders.

Message properties are mapped to message property folders; see Properties specified as MQRFH2 elements. The *NameValueData* syntax includes all the restrictions necessary to map between property names and property folders.

Properties are intended to be accessed by applications using programming interfaces, and not by direct interaction with the physical contents of a message. From Version 7.0, applications can create user properties and property folders using the MQSETMP and MQINQMP functions; see Message properties. It is good programming practice not to read or write `MQRFH2` headers directly, but to use MQSETMP and MQINQMP.

WebSphere MQ also creates special property folders, such as `<jms></jms>` or `<mqps></mqps>`. It creates special property folders when an application uses a particular programming interface such as the JMS client or the integrated publish/subscribe interface. Most of these special folders also existed in Version 6.0.

MQI applications written for Version 6.0 have no property functions to create message properties directly. However, by placing name/value pairs in a special property folder, such as `<usr></usr>`, an application can create message properties. JMS client applications can also create message properties by setting JMS message properties.

The distinction between property folders and ordinary folders is important. In processing message properties, WebSphere MQ modifies the contents of `MQRFH2` folders. The modifications can affect both property and ordinary folders. Experience with using `MQRFH2` headers has led some programmers to assume wrongly that `MQRFH2` contents are not touched by WebSphere MQ. If an `MQRFH2` contains properties, then in modifying the contents of property folders, WebSphere MQ might alter the position or order of ordinary folders.

For example, in Version 6.0, some applications relied on the relative position or ordering of name/value pairs within a *NameValueData* buffer. By making simplifying assumptions about the physical layout of *NameValueData*, the code in those applications to access name/value pairs is faster, but not so robust. Other applications have treated the folder syntax as XML, whereas the syntax is more restrictive. In practice, additional XML data has largely been preserved. As long as WebSphere MQ did not touch the folder, the additional XML syntax that is invalid in an `MQRFH2`, is not detected. Before WebSphere MQ started to use the `MQRFH2` for message properties, these applications might have worked satisfactorily, as on some platforms the `MQRFH2` folder was not touched.

The `MQRFH2` syntax was, and is, only checked as much as is necessary for the queue manager to process the `MQRFH2`. As WebSphere MQ makes more use of the contents of an `MQRFH2`, for example to process properties, more syntax checking has become necessary. The additional syntax checking can cause previously working applications to behave differently, or to fail. The additional processing affects the contents of properties folders more than the contents of ordinary folders.

In Version 7.5, you can set a new **`PROPCTL`** queue attribute value, `V6COMPAT`, on both the sender and receiver queue, and intervening transmission queues. The effect is to restore `MQRFH2` to its Version 6.0 behavior. Use this option to solve migration problems, and not to simplify parsing `MQRFH2` folders. To make programming MQRFH2 folders simpler, use the message properties MQI. It is also a poor optimization to set `V6COMPAT` so that an application can gain reduced path length from simpler parsing of `MQRFH2` folders. This gain is more than offset by the additional storage and pathlength required by the queue manager to implement `V6COMPAT`.

A Program Temporary Fix (PTF) for z/OS, was created in Version 7.0.1 to disable property processing. This PTF was distributed by IBM service in response to any special request for the fix to solve migration problems. It is superseded by the **PROPCTL** V6COMPAT option.

Related information:

DEFINE queues

NameValueData (MQCHARn)

PROPCTL queue attribute default setting

The default setting of the **PROPCTL** attribute for queues migrated from Version 6.0 is changed from COMPAT to V6COMPAT. The change applies to local, alias, and model queue definitions. The change does not affect remote queues, which have no **PROPCTL** attribute. It does affect transmission queues, because they are local queues.

The change solves the problems experienced by some applications that create MQRFH2 headers when they send messages to applications running on Version 7.0 or Version 7.5. A few applications that rely to some extent on the physical format of MQRFH2 headers, or add additional attributes to MQRFH2 folder tags might experience problems.

The PROPCTL attribute is a property of local, alias, and model queues. It was added to queues in Version 7.0. The default setting of the **PROPCTL** attribute for system queues, such as SYSTEM.DEFAULT.LOCAL.QUEUE or SYSTEM.DEFAULT.ALIAS.QUEUE is COMPAT. If a new queue is defined without setting the LIKE property, it inherits the attributes defined on the corresponding system default queue.

Existing queues that were created before Version 7.0 did not have a PROPCTL attribute. When they are migrated from Version 6.0 to Version 7.0, or Version 7.0.1, they inherit the default setting of the PROPCTL attribute, COMPAT.

In Version 7.5, the default setting of the **PROPCTL** attribute for system queues created with a newly defined queue manager remains COMPAT. However, queues migrated from Version 6.0 behave differently. The **PROPCTL** attribute is set to V6COMPAT on all queues. The behavior of the **PROPCTL** attribute is shown in Figure 100 on page 630.

In the first row, QM1 is migrated directly from Version 6.0 to Version 7.5. The **PROPCTL** queue attribute is set to the default value V6COMPAT. Because the **PROPCTL** queue attribute is also set to V6COMPAT on all the default queues, newly created queues, shown in black, also have the **PROPCTL** queue attribute set to V6COMPAT.

In the second row, QM2 is migrated first to Version 7.0, or Version 7.0.1, and then to Version 7.5. The migration to Version 7.0 sets the **PROPCTL** queue attribute to COMPAT. Because the **PROPCTL** queue attribute is also set to COMPAT on all the default queues, newly created queues, shown in black, also have the **PROPCTL** queue attribute set to COMPAT. When QM2 is migrated to Version 7.5, all queues already have a **PROPCTL** attribute, and so they retain the same setting. If the setting on the all the default queues has been left as COMPAT, new queues also have the **PROPCTL** queue attribute set to COMPAT.

In the third row, QM3 is created in Version 7.0 or Version 7.0.1. The default setting of the **PROPCTL** queue attribute is COMPAT. When QM3 is migrated to Version 7.5, the queues retain their **PROPCTL** setting.

In the fourth row, a new queue manager is created in Version 7.5. Like Version 7.0, the default setting of the **PROPCTL** queue attribute is COMPAT.

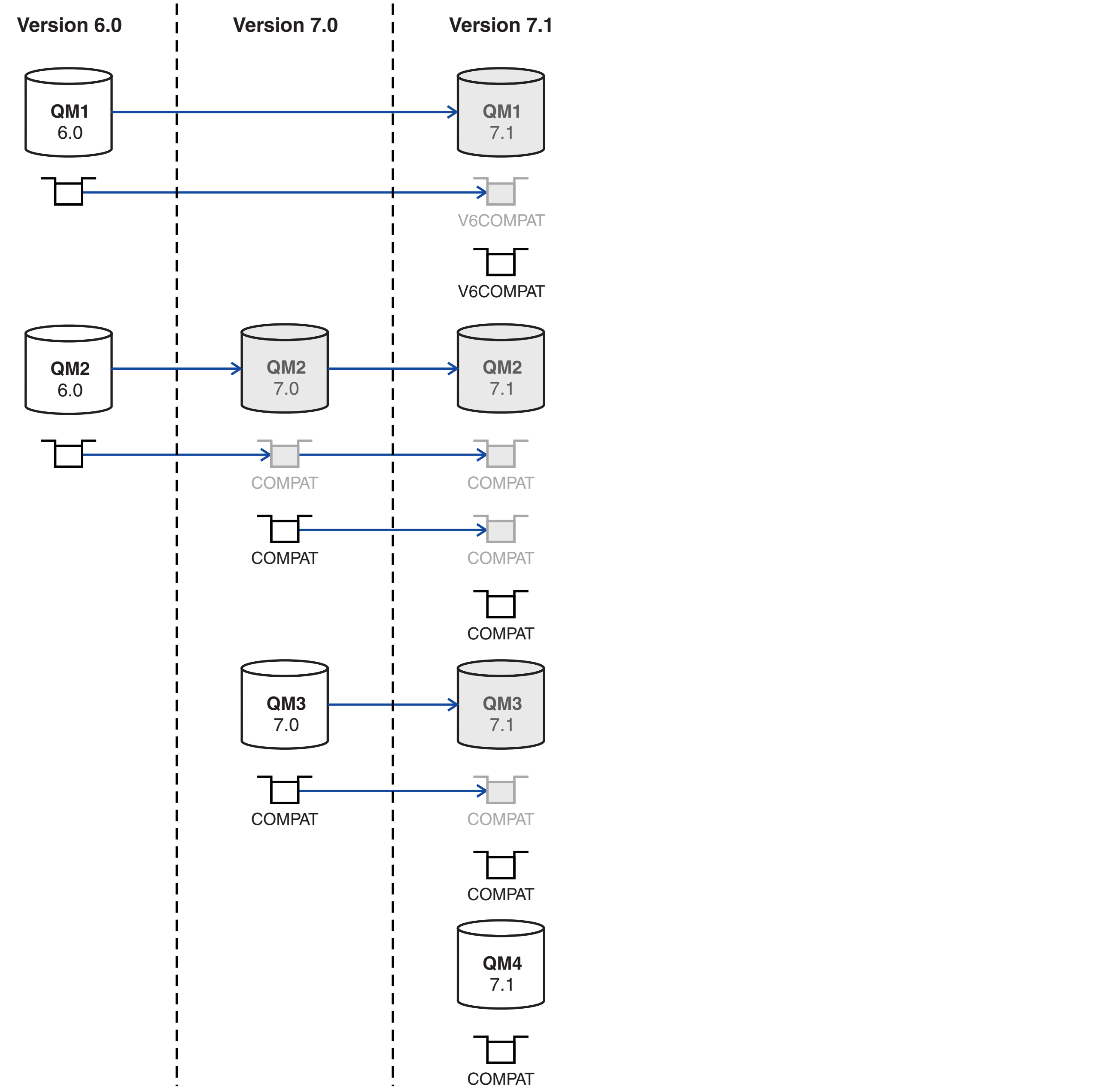


Figure 100. Migration and creation of queues with default PROPCTL attribute setting

This change affects not only existing queues that were created on Version 6.0, but also effects new queues created in Version 7.5. New queues are effected because default system queues, such as `SYSTEM.DEFAULT.LOCAL.QUEUE` and `SYSTEM.DEFAULT.ALIAS.QUEUE` are also migrated to Version 7.5, and their **`PROPCTL`** property is set to `V6COMPAT`. Unless you override the default setting of **`PROPCTL`** when creating a queue, the definition inherits the **`PROPCTL`** attribute value from a system default queue.

The intended consequence of this change is that queues defined on a queue manager migrated from Version 6.0 behave differently to queues defined on a queue manager created on Version 7.0 or later. The change does not apply to migration from Version 6.0 to Version 7.0 or Version 7.0.1. A queue manager that is migrated directly to Version 7.5 from Version 6.0 behaves differently to a queue manager migrated indirectly from Version 6.0.

V6COMPAT does not change the behavior of Version 6.0 applications migrated to Version 7.5 that would have worked correctly with the previous default value of COMPAT. The new setting resolves a migration problem for a few programs that rely of how the MQRFH2 header is formatted in Version 6.0. For new applications, V6COMPAT has a small impact on the performance of message transfer.

If you want to override the change on the system default queues, after migrating a queue manager to Version 7.5, redefine the PROPCTL attribute on the following queues:

SYSTEM.DEFAULT.ALIAS.QUEUE

SYSTEM.DEFAULT.LOCAL.QUEUE

SYSTEM.DEFAULT.MODEL.QUEUE

SYSTEM.DURABLE.MODEL.QUEUE

SYSTEM.DURABLE.SUBSCRIBER.QUEUE

SYSTEM.NDURABLE.MODEL.QUEUE

SYSTEM.DEAD.LETTER.QUEUE

SYSTEM.CLUSTER.TRANSMIT.QUEUE

All queues migrated from Version 6.0, including system queues have **PROPCTL** set to V6COMPAT. The PROPCTL setting on the other system queues does not affect the behavior of the queue manager. The **PROPCTL** setting has no effect on programs that set an explicit MQGMO_PROPORTIES_* option.

Related tasks:

"Migrating an application that uses the MQRFH2 folder" on page 526
If you run an application that uses MQRFH2 folders in an WebSphere MQ message, this migration task might be relevant to you. You might find, in a few cases, that the results of parsing an MQRFH2 are different inVersion 7.5 to Version 6.0. Version 7.5 uses the MQRFH2 folder to transport message properties. As a result, in some cases it reformats MQRFH2 folders. The default setting of the PROPCTL queue and channel attribute eliminates most cases of incompatibility, but not all.

"Parsing a message containing MQRFH2 headers correctly" on page 531
Writing a procedure to parse MQRFH2 headers correctly is difficult. Use the syntax reference information and the hints in this topic to guide you. If at all possible, use the message property MQI and message properties rather than MQRFH2 folders and name/value pairs to transfer additional message properties. If you use the message properties interface, you remove the need to read, write, and parse an MQRFH2 header at all.

"Using the GMO_PROPERTIES_* option to control message properties" on page 530
Set the GMO_PROPERTIES_* options in a new or changed MQI application connected to a Version 7.5 queue manager to control how properties are returned to the application.

"Using the **PROPCTL** channel property to control message properties" on page 528
The task guides you on setting the **PROPCTL** channel property for different types of channel. **PROPCTL** controls the transmission of message properties. **PROPCTL** can prevent an application connected to a V6 queue manager from receiving V7.0 message properties that it cannot process correctly.

"Using the **PROPCTL** queue property to control message properties" on page 529
Set the **PROPCTL** attribute on a queue to control which properties of a message are returned to an application that gets a message from the queue. Prevent an application migrated from V6 from receiving unexpected message properties in messages retrieved from the queue. Stage the conversion of applications to use the message property interface, instead of accessing an MQRFH2 header directly.

Related reference:

"MQGMO message property option settings" on page 734
Use MQGMO message property options to control how message properties are returned to an application.

"**PROPCTL** channel options" on page 730
Use **PROPCTL** channel attribute to control which message properties are included in a message that is sent from a Version 7.5 queue manager to a queue manager from an earlier version of WebSphere MQ.

"**PROPCTL** queue options" on page 731
Use the **PROPCTL** queue attribute to control how message properties are returned to an application that calls MQGET without setting any MQGMO message property options.

Related information:

PROPCTL

Rules for selecting the WebSphere MQ messaging provider mode

The WebSphere MQ messaging provider has two modes of operation: WebSphere MQ messaging provider normal mode, and WebSphere MQ messaging provider migration mode. You can select which mode a JMS application uses to publish and subscribe.

The WebSphere MQ messaging provider normal mode uses all the features of a MQ queue manager to implement JMS. This mode is used only to connect to a WebSphere MQ queue manager and can connect to queue managers in either client or bindings mode. This mode is optimized to use the new function.

If you are not using WebSphere MQ Real-Time Transport, the mode of operation used is determined primarily by the **PROVIDERVERSION** property of the connection factory. If you cannot change the connection factory that you are using, you can use the `com.ibm.msg.client.wmq.overrideProviderVersion` property to override any setting on the connection factory. This override applies to all connection factories in the JVM but the actual connection factory objects are not modified.

You can set **PROVIDERVERSION** to the possible values: `7`, `6`, or *unspecified*. However, **PROVIDERVERSION** can be a string in any one of the following formats:

- V.R.M.F
- V.R.M
- V.R
- V

where V, R, M and F are integer values greater than or equal to zero.

7 - Normal mode

Uses the WebSphere MQ messaging provider normal mode.

If you set **PROVIDERVERSION** to 8 only the WebSphere MQ messaging provider normal mode of operation is available. If the queue manager specified in the connection factory settings is not a Version 7.0.1, or later, queue manager, the createConnection method fails with an exception.

The WebSphere MQ messaging provider normal mode uses the sharing conversations feature and the number of conversations that can be shared is controlled by the **SHARECNV()** property on the server connection channel. If this property is set to `0`, you cannot use WebSphere MQ messaging provider normal mode and the createConnection method fails with an exception.

6 - Migration mode

Uses the WebSphere MQ messaging provider migration mode.

The WebSphere MQ classes for JMS use the features and algorithms supplied with WebSphere MQ version 6.0. If you want to connect to WebSphere Event Broker version 6.0 or WebSphere Message Broker version 6.0 or 6.1 using WebSphere MQ Enterprise Transport version 6.0, you must use this mode. You can connect to a WebSphere MQ version 7.0.1 queue manager using this mode, but none of the new features of a version 7.0.1 queue manager are used, for example, read ahead or streaming. If you have a WebSphere MQ version 7.0.1 client connecting to a WebSphere MQ version 7.0.1 queue manager on a distributed platform or a WebSphere MQ version 7.0.1 queue manager on z/OS, then the message selection is done by the queue manager rather than on the client system.

unspecified

The **PROVIDERVERSION** property is set to *unspecified* by default.

A connection factory that was created with a previous version of WebSphere MQ classes for JMS in JNDI takes this value when the connection factory is used with the new version of WebSphere MQ classes for JMS. The following algorithm is used to determine which mode of operation is used. This algorithm is used when the createConnection method is called and uses other aspects of the connection factory to determine if WebSphere MQ messaging provider normal mode or WebSphere MQ messaging provider migration mode is required.

1. First, an attempt to use WebSphere MQ messaging provider normal mode is made.
2. If the queue manager connected is not WebSphere MQ version 7.0.1, or later, the connection is closed and WebSphere MQ messaging provider migration mode is used instead.
3. If the **SHARECNV** property on the server connection channel is set to 0, the connection is closed and WebSphere MQ messaging provider migration mode is used instead.
4. If **BROKERVER** is set to V1 or the new default *unspecified* value, WebSphere MQ messaging provider normal mode continues to be used, and therefore any publish/subscribe operations use the new WebSphere MQ version 7.0.1, or later, features.

   See ALTER QMGR for information about the PSMODE parameter of the ALTER QMGR command for further information on compatibility.
5. If **BROKERVER** is set to V2 the action taken depends on the value of **BROKERQMGR** :
   - If the **BROKERQMGR** is blank:

     If the queue specified by the **BROKERCONQ** property can be opened for output (that is, MQOPEN for output succeeds) and **PSMODE** on the queue manager is set to COMPAT or DISABLED, then WebSphere MQ messaging provider migration mode is used.
   - If the queue specified by the **BROKERCONQ** property cannot be opened for output, or the **PSMODE** attribute is set to ENABLED:

     WebSphere MQ messaging provider normal mode is used.
   - If **BROKERQMGR** is non-blank :

     WebSphere MQ messaging provider mode is used.

Related information:

When to use PROVIDERVERSION
In two cases you must override the default selection of **PROVIDERVERSION** for the WebSphere MQ classes for JMS to work correctly.

BROKERQMGR

BROKERCONQ

PSMODE

Publish/subscribe changes

Publish/subscribe function is now performed by the queue manager, rather than by a separate publish/subscribe broker. When you upgrade a IBM WebSphere MQ Version 6.0 queue manager, publish/subscribe function is not automatically migrated. You must migrate publish/subscribe using the **strmqbrk** command. After migration, with the differences noted in the following topics, existing Version 6.0 publish/subscribe programs work without change.

The programs themselves are not migrated; Version 6.0 publish/subscribe programs work when connected to a later version queue manager, after migrating publish/subscribe. The publish/subscribe integrated into a queue manager emulates a Version 6.0 publish/subscribe broker. Old and new publish/subscribe programs can work along side one another. They can share the same topic space on the same queue manager.

Version 6.0 and later version queue managers can exchange publications and subscriptions in a publish/subscribe hierarchy.

Publish/Subscribe: Altering fields in a subscription:

In Version 6.0, you can update the topic, destination queue, and subscription name of a subscription as long as you have named the subscription. You can no longer alter these fields, once a subscription has been created.

Related information:

Changing the registration of an application

MQSO_ALTER

Publish/Subscribe: Attributes:

Attributes that control publish/subscribe are now integrated into the queue manager. Some are equivalent to attributes controlling Version 6.0 publish/subscribe.

Table 76. Queue manager publish/subscribe attributes

Queue manager attribute	Description	Version 6.0 equivalent
PSMODE	Controls whether the publish/subscribe engine and the queued publish/subscribe interface are running.	**strmqbrk**, **endmqbrk**
PSNPMSG	Controls whether to continue processing after a failure to process a non-persistent message.	DiscardNonPersistentInputMsg
PSNPRES	Controls whether to continue processing after a failure to deliver a response message.	PubSubNPResponse
PSRTYCNT	Controls how many times to attempt to process a publish/subscribe command message	MaxMsgRetryCount
PSSYNCPT	Controls whether to process queued publish/subscribe messages under sync point control	SyncPointIfPersistent

Related information:

Working with queue managers

Publish/Subscribe: Application migration:

The Version 6.0 IBM WebSphere MQ publish/subscribe command message interface has been superseded by the IBM WebSphere MQ publish/subscribe programming interface. It is your choice whether to migrate existing applications to the new interface. The following topics provide reference information to help you migrate applications to the publish/subscribe programming interface.

The migration is not a straightforward mapping: the publish/subscribe programming models are different. To maintain compatibility you can run both kinds of publish/subscribe programs, and they work with the same publication and subscriptions.

You might have migrated the publish/subscribe broker in WebSphere Message Broker or WebSphere Event Broker to IBM WebSphere MQ. If so, you might be running applications using two different versions of the publish/subscribe command message interface. One version uses commands formatted as `MQRFH` headers, and the other as `MQRFH2` headers. Both versions of the command message interface are supported along with the programming interface. You can write publish/subscribe programs for WebSphere Message Broker version 7.0 using either of the command message interfaces, or the programming interface.

The publish/subscribe programming interface is integrated into the MQI. It is sometimes known as integrated publish/subscribe to distinguish it from queued publish/subscribe. Queued publish/subscribe is name given to the implementation of the Version 6.0 publish/subscribe command interface. You can use both queued and integrated publish/subscribe, and use them together with the same topics and subscriptions.

JMS can use either interface. In most circumstances, JMS defaults to using the integrated interface. You can control which interface it uses by setting the JMS property **PROVIDERVERSION** to 6 or 7.

Related information:

Writing queued publish/subscribe applications

Writing integrated publish/subscribe applications

Publish/Subscribe Application migration: Identity:

You can identify a subscriber in two ways in Version 6.0. All subscribers have a traditional identity, and you can add a subscription name. All JMS subscriptions must have a subscription name. In Version 7.5, all subscriptions have only a subscription name. In Version 6.0, the traditional identity is also used to identify a publisher. In Version 7.5 publishers do not have an identity, except for the connection they establish with the queue manager.

The traditional identity is a combination of queue name, queue manager name, and optional correlation identifier.

Table 77. Effects of Identity changes

Version 6.0	**Version 7.5**
The subscription name is associated with a traditional identity by specifying addname, and a subscription name, on a **Register Subscriber** command.	The subscription name is the SubName field in the Subscription Descriptor, MQSD. You must set the SubName field in the Subscription Descriptor, MQSD when issuing MQSUB to identify a subscriber.
The correlation identifier has a secondary use which is to allow subscribers to MQGET by CorrelId and get publications only for a particular subscription. If multiple subscriptions are using same queue, the CorrelId differentiated the subscriptions and publications.	You can optionally set the SubCorrelId in the Subscription Descriptor, MQSD, in the MQSUB call. Alternatively the queue manager generates the value of SubCorrelId. **Note:** The correlation identifier is not passed between queue managers in a broker hierarchy.
You can set the registration option MQPS_ANONYMOUS, "Anon". The option tells the broker that the identity of the publisher is not to be divulged, except to subscribers with additional authority.	A publisher can stop subscribers replying to publications they receive. Set the MQPMO_SUPPRESS_REPLYTO **MQPMO** option when publishing.
A publisher can set MQPS_OTHER_SUBSCRIBERS_ONLY, "OtherSubsOnly", and based on the identity of the publisher and subscriber avoid sending publications to a subscriber who is also the publisher.	Publishing and subscribing applications can be identified by their connection to the queue manager. The same connection can be used to both publish and subscribe. If it is, you can prevent publications created on the connection being sent to a subscription on the same connection. Set the subscription option, MQSO_NOT_OWN_PUBS.

Related information:

Publish

Publisher and subscriber identity

Publisher identity

Register Publisher

Register Subscriber

SubCorrelId (MQBYTE24)

SubName (MQCHARV)

Subscription options

Publish/Subscribe Application migration: Stream name replaced by topic object:

Stream names are not used in the Version 7.5 integrated publish/subscribe interface. Topic objects take the place of Version 6.0 stream names.

You can configure Version 7.5 to emulate streams if queued and integrated publish/subscribe must coexist; see the related information.

Table 78. Migrating streams to topics

Version 6.0	Version 7.5
The optional command message parameter, `MQPS_STREAM_NAME`, sets a stream name in the version 6.0 publish/subscribe interface.	Before issuing MQPUT to publish to a topic, or MQSUB to subscribe, set the `ObjectName` and `ObjectString` fields. The `ObjectName` field is equivalent to a stream name, and `ObjectString` to the topic string. `ObjectName` is the name of an administrative topic object. To publish, set the fields in the Object Descriptor, `MQOD`; to subscribe, set the fields in the Subscription Descriptor, MQSD.
	To map the topic object to an emulated stream; see the related references.
If a stream does not exist the publish/subscribe broker returns `MQRCCF_STREAM_ERROR` to the application.	If a topic object does not exist the queue manager returns `MQRC_UNKNOWN_OBJECT_NAME` to the application.

Table 78. Migrating streams to topics (continued)

Version 6.0	Version 7.5
As an example, a version 6.0 application publishes to a stream queue called SYSTEM.BROKER.RESULTS.STREAM with a topic string of Sport/Soccer/State/LatestScore. 1. Create a stream queue called SYSTEM.BROKER.RESULTS.STREAM. 2. The application publishes to the topic string Sport/Soccer/State/LatestScore using the stream SYSTEM.BROKER.RESULTS.STREAM.	In Version 7.5, the application publishes to a topic SYSTEM.BROKER.RESULTS.STREAM with a topic string of Sport/Soccer/State/LatestScore. 1. Create a topic object called SYSTEM.BROKER.RESULTS.STREAM; define it with a topic string of RESULTS.STREAM. 2. The application publishes to the topic string Sport/Soccer/State/LatestScore using the topic object SYSTEM.BROKER.RESULTS.STREAM. The resulting publication has a topic string of RESULTS.STREAM/Sport/Soccer/State/LatestScore. If the application must preserve the same topic string, then define the topic SYSTEM.BROKER.RESULTS.STREAM with a topic string of "/". If you do so, you lose the ability to distinguish topics published to different streams.
	You can extend the example, and emulate the version 6.0 stream in Version 7.5. With the change, the version 6 and Version 7.5 applications can interoperate. 1. Set the **WILDCARD** property of the SYSTEM.BROKER.RESULTS.STREAM topic to BLOCK. 2. Create a queue called SYSTEM.BROKER.RESULTS.STREAM 3. Add SYSTEM.BROKER.RESULTS.STREAM to the namelist SYSTEM.QPUBSUB.QUEUE.NAMELIST.

Related reference:

"Publish/Subscribe: Streams" on page 655
Streams are used in Version 6.0 to isolate publish/subscribe applications. Streams are now emulated. Migration from Version 6.0 streams is automatic. Existing Version 6.0 applications that use streams continue to work unchanged when migrated to, or interoperating with, later versions of IBM WebSphere MQ. In some unusual cases, you might need to modify the results of the automatic migration. If you introduce new streams to Version 6.0 applications, you must configure the emulated streams manually.

Related information:

Adding a stream, version 7.0.1
You can add streams manually so that they coexist with streams migrated from Version 6.0 queue managers.

Streams, Version 6

Stream and topics, version 7.0.1
Queued publish/subscribe has the concept of a publication stream that does not exist in the integrated publish/subscribe model. In queued publish/subscribe, streams provide a way of separating the flow of information for different topics. In IBM WebSphere MQ Version 6.0, a stream is implemented as a queue, defined at each broker that supports the stream. Each queue has the same name (the name of the stream). From IBM WebSphere MQ Version 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

Writing publisher applications, version 7.0.1

Publish/Subscribe Application migration: **Delete Publication** *replaced by* **ClearTopic***:*

Replace **Delete Publication** by the PCF **Clear Topic** command. Migrate the **Delete Publication** parameters and options to the equivalent options in the **ClearTopic** command.

Table 79. Migrate **Delete Publication** *to* **Clear Topic**

Delete Publication	**Clear Topic**
MQPS_COMMAND with value MQPS_DELETE_PUBLICATION	Implicit in the **ClearTopic** command
MQPS_TOPIC	TopicString parameter(Repeat the **Clear Topic** command if **Delete Publication** specifies multiple topics.)
MQPS_DELETE_OPTIONS	ClearType parameter
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636

Table 80. Return code migration

Reason code returned by the publish/subscribe broker	**Reason code returned by the queue manager**
MQRCCF_STREAM_ERROR	MQRC_UNKNOWN_OBJECT_NAME
MQRCCF_TOPIC_ERROR	MQRC_OBJECT_STRING_ERROR
MQRCCF_INCORRECT_STREAM	A stream error can no longer occur.

Related information:

Clear Topic

Delete Publication

Delete Publication (Format)

Publish/Subscribe Application migration: **Deregister Publisher** *replaced by MQCLOSE:*

Replace the **Deregister Publisher** command by the MQCLOSE Message Queue Interface (MQI) call. Migrate the **Deregister Publisher** parameters and options to the equivalent options in the MQCLOSE call.

Deregister Publisher has a different lifecycle to MQCLOSE. Between **Register Publisher** and **Deregister Publisher** a publisher remains registered, even when the publisher is not connected. Integrated publish/subscribe has no **Register Publisher** equivalent. MQCLOSE releases the object connected by MQOPEN. When the publisher application is not connected, the publisher is not known by the queue manager. A connection can also be closed by the queue manager without the publisher application issuing MQCLOSE.

Table 81. Migrate **Deregister Publisher** *to MQCLOSE.*

Deregister Publisher	**MQCLOSE**
MQPS_COMMAND with value MQPS_DEREGISTER_PUBLISHER	"Deregistering" is implicit in releasing the handle to a topic previously opened using MQOPEN with the MQOO_OUTPUT option.
Queue and queue manager name to publish to. The names are provided either by MQPS_Q_MGR_NAME and MQPS_Q_NAME, or by the ReplyToQMgr and ReplyToQ fields in the Message Descriptor, MQMD, of the command message.	Publications are sent to topics, not queues. The topic is identified by the ObjectName and ObjectString of the topic Object Descriptor, MQOD. The queue manager is the queue manager the application is connected to.
MQPS_REGISTRATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQREGO_CORREL_ID_AS_ IDENTITY (MQLONG)	Publishers have no lasting identity; see "Publish/Subscribe Application migration: Identity" on page 635
MQPS_REGISTRATION_OPTIONS = MQPS_DEREGISTER_ALL (MQCHAR*) or MQREGO_DEREGISTER_ ALL (MQLONG)	Similar to issuing MQDISC, and implicitly closing all open topic connections.

Table 81. Migrate **Deregister Publisher** *to MQCLOSE. (continued)*

Deregister Publisher	**MQCLOSE**
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636

Table 82. Migrate return codes

Reason code returned by the publish/subscribe broker	**Reason code returned by the queue manager**
MQRCCF_STREAM_ERROR	MQRC_HOBJ_ERROR
MQRCCF_TOPIC_ERROR	MQRC_HOBJ_ERROR
MQRCCF_NOT_REGISTERED	MQRC_HOBJ_ERROR
MQRCCF_Q_MGR_NAME_ERROR	MQRC_HOBJ_ERROR
MQRCCF_Q_NAME_ERROR	MQRC_HOBJ_ERROR
MQRCCF_DUPLICATE_IDENTITY	MQRC_HOBJ_ERROR
MQRCCF_UNKNOWN_STREAM	MQRC_HOBJ_ERROR
MQRCCF_REG_OPTIONS_ERROR	MQRC_OPTIONS_ERROR

Related information:

Deregister Publisher

Deregister Publisher (Format)

MQCLOSE

MQDISC

MQOD

MQOPEN

Publish/Subscribe Application migration: **Deregister Subscriber** *replaced by MQCLOSE:*

Replace the **Deregister Subscriber** command by the MQCLOSE Message Queue Interface (MQI) call. Migrate the **Deregister Subscriber** parameters and options to the equivalent options in the MQCLOSE call.

Deregister Subscriber has a different lifecycle to MQCLOSE. **Register Subscriber** might be issued by a different application or process to **Deregister Subscriber**. If **Deregister Subscriber** is used in a different program to **Register Subscriber**, call MQSUB with the MQSO_RESUME option to get a handle to the subscription.

Table 83. Migrate **Deregister Subscriber** *to MQCLOSE*

Deregister Subscriber	**MQCLOSE**
MQPS_COMMAND with value MQPS_DEREGISTER_SUBSCRIBER	Call MQCLOSE with the option MQCO_REMOVE_SUB
Queue and queue manager name to subscribe to. The names are provided either by MQPS_Q_MGR_NAME and MQPS_Q_NAME, or by the ReplyToQMgr and ReplyToQ fields in the Message Descriptor, MQMD, of the command message.	A connection to a subscription also has a connection to the queue to which matching publications are sent. A handle to the queue is set or returned on the MQSUB call. Whether a handle is set or returned depends on whether the queue is managed or unmanaged, and whether the subscription is being created or resumed. The queue manager is the queue manager the application is connected to.

Table 83. Migrate **Deregister Subscriber** *to MQCLOSE (continued)*

Deregister Subscriber	**MQCLOSE**
MQPS_REGISTRATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQREGO_CORREL_ID_AS_IDENTITY (MQLONG)	The subscriber identity is implicit in the connection handle obtained by MQSUB. It is released by MQCLOSE. The correlation identifier is no longer part of a subscription identity.
MQPS_REGISTRATION_OPTIONS = MQPS_DEREGISTER_ALL (MQCHAR*) or MQREGO_DEREGISTER_ALL (MQLONG)	Only one topic is subscribed to in one MQSUB call. Releasing the handle with MQCLOSE, closes the only subscription. If multiple subscriptions are open, MQDISC closes them all.
MQPS_REGISTRATION_OPTIONS = MQPS_FULL_RESPONSE (MQCHAR*) or MQREGO_FULL_RESPONSE (MQLONG)	MQSUB returns all the response fields in the MQSD structure.
MQPS_REGISTRATION_OPTIONS = MQPS_LEAVE_ONLY (MQCHAR*) or MQREGO_LEAVE_ONLY (MQLONG)	Subscriptions no longer have subidentities, so there is no equivalent to the MQPS_LEAVE_ONLY option.
MQPS_REGISTRATION_OPTIONS = MQPS_VARIABLE_USER_ID (MQCHAR*) or MQREGO_VARIABLE_USER_ID (MQLONG)	Subscriptions are no longer shared between subscriber identities. Deregistering a subscription for another subscriber has no direct equivalent.
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636
MQPS_SUBSCRIPTION_NAME	The subscription is identified by the SubName field of the Subscription Descriptor, MQSD.
MQPS_TOPIC	The topic string is identified by the ObjectString field in the Subscription Descriptor, MQSD.

Table 84. Migrate return codes

Reason code returned by the publish/subscribe broker	**Reason code returned by the queue manager**
MQRCCF_STREAM_ERROR	MQRC_HOBJ_ERROR
MQRCCF_TOPIC_ERROR	MQRC_HOBJ_ERROR
MQRCCF_NOT_REGISTERED	MQRC_HOBJ_ERROR
MQRCCF_Q_MGR_NAME_ERROR	MQRC_HOBJ_ERROR
MQRCCF_Q_NAME_ERROR	MQRC_HOBJ_ERROR
MQRCCF_DUPLICATE_IDENTITY	MQRC_HOBJ_ERROR
MQRCCF_UNKNOWN_STREAM	MQRC_HOBJ_ERROR
MQRCCF_REG_OPTIONS_ERROR	MQRC_OPTIONS_ERROR

Related information:

Deregister subscriber

Deregister subscriber (format)

MQCLOSE

MQDISC

MQOD

MQOPEN

MQSD - Subscription descriptor

MQSUB

Publish/Subscribe Application migration: **Publish** *replaced by MQPUT and MQPUT1:*

Replace the **Publish** command by the MQPUT or MQPUT1 Message Queue Interface (MQI) calls. Migrate the **Publish** parameters and options to the equivalent options in the MQPUT calls.

Table 85. Migrate **Publish** *to MQPUT and MQPUT1*

Publish	**MQPUT or MQPUT1**
MQPS_COMMAND with value MQPS_PUBLISH	Call MQPUT or MQPUT1 to an object handle obtained by opening a topic for MQOO_OUTPUT.
MQPS_ANONYMOUS or MQREGO_ANONYMOUS	There is no equivalent to registering a publisher for new applications and registering it as an anonymous publisher. To achieve the same result of not passing a replyto destination to subscribers, call MQPUT or MQPUT1 setting the **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_TOPIC	Set ObjectString in the Object Descriptor, MQOD, of the publication topic to the topic string. A **Publish** command can publish to more than one topic. To achieve the same result, an application must call MQOPEN call for each topic. It must then call MQPUT for every topic, sending each the same publication. Alternatively, combine the two calls, by using MQPUT1.
MQPS_PUBLICATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQPUBO_CORREL_ID_AS_IDENTITY (MQLONG)	Publishers have no lasting identity; see "Publish/Subscribe Application migration: Identity" on page 635
MQPS_PUBLICATION_OPTIONS = MQPS_IS_RETAINED_ PUBLICATION (MQCHAR*) or MQPUBO_IS_RETAINED_ PUBLICATION (MQLONG)	The message property, MQIsRetained, mqps.Ret, is set on a retained publication sent to a subscriber. A subscriber can also request no retained publications, by setting the subscription option, MQSO_NEW_PUBLICATIONS_ONLY, and only a retained publication, by setting MQSO_PUBLICATIONS_ON_REQUEST
MQPS_PUBLICATION_OPTIONS = MQPS_NO_ REGISTRATION (MQCHAR*) or MQPUBO_NO_ REGISTRATION (MQLONG)	This option is not relevant to integrated publish/subscribe, because publishers are not registered.
MQPS_PUBLICATION_OPTIONS = MQPS_OTHER_SUBSCRIBERS_ONLY (MQCHAR*) or MQPUBO_OTHER_SUBSCRIBERS_ONLY (MQLONG)	If an application is not to receive its own publications, subscribe using the option MQPMO_NOT_OWN_SUBS on the MQPUT or MQPUT1 calls. **Note:** The scope of the same publisher or subscriber is a common connection handle to the queue manager.
MQPS_PUBLICATION_OPTIONS = MQPS_RETAIN_PUBLICATION (MQCHAR*) or MQPUBO_RETAIN_PUBLICATION (MQLONG)	Call MQPUT or MQPUT1 setting the **MQPMO** option MQPMO_RETAIN.
MQPS_INTEGER_DATA (MQCHAR*) or MQPUBO_INTEGER_DATA (MQLONG), or MQPS_STRING_DATA (MQCHAR*) or MQPUBO_STRING_DATA (MQLONG)	Set MQPubStrIntData, mqpse.Sid; see Publish/subscribe message properties.
MQPS_Q_MGR_NAME	The ReplyToQMgr field in the Message Descriptor, MQMD, of the publication contains the queue manager the publisher is connected to. The field is blank if the publisher sets the **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_Q_NAME	The ReplyToQ field in the Message Descriptor, MQMD, of the publication contains the queue the publisher expects replies on. The publisher can leave the ReplyToQ field blank.

Table 85. Migrate **Publish** *to MQPUT and MQPUT1 (continued)*

Publish	**MQPUT or MQPUT1**
MQPS_REGISTRATION_OPTIONS = MQPS_ANONYMOUS (MQCHAR*) or MQREGO_ANONYMOUS (MQLONG)	There is no equivalent to registering a publisher for new applications and registering it as an anonymous publisher. To achieve the same result of not passing a replyto destination to subscribers, call MQPUT or MQPUT1 setting the **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_REGISTRATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQREGO_CORREL_ID_AS_IDENTITY (MQLONG)	Publishers have no lasting identity; see "Publish/Subscribe Application migration: Identity" on page 635
MQPS_REGISTRATION_OPTIONS = MQPS_DIRECT_REQUEST (MQCHAR*) or MQREGO_DIRECT_ EQUEST (MQLONG)	Set ReplyToQ and ReplyToQMgr in the Message Descriptor, MQMD, of a publication message. Do not set **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_REGISTRATION_OPTIONS = MQPS_LOCAL (MQCHAR*) or MQREGO_LOCAL (MQLONG)	Set **MQPMO** option MQPMO_SCOPE_QMGR. **Note:** The meaning of **SCOPE** has changed; see "Publish/Subscribe: Local and global publications and subscriptions" on page 651
MQPS_SEQUENCE_NUMBER	Set MQPubSeqNum, mqpse.Seq
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636

Related information:

Publish

Publish (Format)

MQOD

MQOPEN

MQPMO

MQPUT

MQPUT1

Publish/Subscribe Application migration: **Register Publisher** *replaced by MQOPEN:*

Replace the **Register Publisher** command by the MQOPEN Message Queue Interface (MQI) call. Migrate the **Register Publisher** parameters and options to the equivalent options in the MQPUT calls.

Register Publisher has a different lifecycle to MQOPEN. Between **Register Publisher** and **Deregister Publisher** a publisher remains registered, even when the publisher is not connected. Integrated publish/subscribe has no **Register Publisher** equivalent. MQCLOSE releases the object connected by MQOPEN. When the publisher application is not connected, the publisher is not known by the queue manager. A connection can also be closed by the queue manager without the publisher application issuing MQCLOSE.

Table 86. Migrate **Register Publisher** *to MQOPEN*

Register Publisher	**MQOPEN**
MQPS_COMMAND with value MQPS_REGISTER_PUBLISHER	Call MQPUT or MQPUT1 to an object handle obtained by opening a topic for MQOO_OUTPUT. If your application did not use **Register Publisher**, see "Publish/Subscribe Application migration: **Publish** replaced by MQPUT and MQPUT1" on page 641.
MQPS_TOPIC	Set ObjectString in the Object Descriptor, MQOD, of the publication topic to the topic string. A **Publish** command can publish to more than one topic. To achieve the same result, an application must call MQOPEN call for each topic. It must then call MQPUT for every topic, sending each the same publication. Alternatively, combine the two calls, by using MQPUT1.
MQPS_Q_MGR_NAME	You must set the ReplyToQMgr field in the Message Descriptor, MQMD, for every publication to identify the publisher. The field is blank if the publisher sets the **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_Q_NAME	You must set the ReplyToQ field in the Message Descriptor, MQMD, for every publication to identify the publisher. The publisher can leave the ReplyToQ field blank.
MQPS_REGISTRATION_OPTIONS = MQPS_ANONYMOUS (MQCHAR*) or MQREGO_ANONYMOUS (MQLONG)	There is no equivalent to registering a publisher for new applications and registering it as an anonymous publisher. To achieve the same result of not passing a replyto destination to subscribers, call MQPUT or MQPUT1 setting the **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_REGISTRATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQREGO_CORREL_ID_AS_IDENTITY (MQLONG)	Publishers have no lasting identity; see "Publish/Subscribe Application migration: Identity" on page 635
MQPS_REGISTRATION_OPTIONS = MQPS_DIRECT_REQUEST (MQCHAR*) or MQREGO_DIRECT_ EQUEST (MQLONG)	Set ReplyToQ and ReplyToQMgr in the Message Descriptor, MQMD, of a publication message. Do not set **MQPMO** option MQPMO_SUPPRESS_REPLYTO.
MQPS_REGISTRATION_OPTIONS = MQPS_LOCAL (MQCHAR*) or MQREGO_LOCAL (MQLONG)	Set **MQPMO** option MQPMO_SCOPE_QMGR. **Note:** The meaning of **SCOPE** has changed; see "Publish/Subscribe: Local and global publications and subscriptions" on page 651
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636

Related information:

Register Publisher

Register Publisher (Format)

MQOD

MQOPEN

MQPMO

MQPUT

MQPUT1

Publish/Subscriber Application migration: **Register Subscriber** *replaced by MQSUB:*

Replace the **Register Subscriber** command by the MQSUB Message Queue Interface (MQI) call. Migrate the **Register Subscriber** parameters and options to the equivalent options in the MQSUB calls.

Table 87. Migrate **Register Subscriber** *to MQSUB*

Register Subscriber	**MQSUB**
MQPS_COMMAND with value MQPS_REGISTER_SUBSCRIBER	Call MQSUB. If your application did not use **Register Subscriber** then the use of the MQSUB verb is not required for equivalent behavior.
MQPS_TOPIC	Set ObjectString in the Subscription Descriptor, MQSD A **Register Subscriber** command can subscribe to more than one topic. To achieve the same result, an application must call MQSUB call for each topic.
MQPS_Q_MGR_NAME	MQSUB requires the connection handle of the queue manager to which the application is connected.
MQPS_Q_NAME	MQSUB either returns the queue handle of the managed queue or the application can provide the handle to an unmanaged queue. Publications that match the subscription are returned to the queue.
MQPS_REGISTRATION_OPTIONS = MQPS_ADD_NAME (MQCHAR*) or QREGO_ADD_NAME (MQLONG)	A subscription has only single name.
MQPS_REGISTRATION_OPTIONS = MQPS_ANONYMOUS (MQCHAR*) or MQREGO_ANONYMOUS (MQLONG)	Not applicable to subscribers.
MQPS_REGISTRATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQREGO_CORREL_ID_AS_IDENTITY (MQLONG)	The subscriber identity is implicit in the connection handle obtained by MQSUB. It is released by MQCLOSE. The correlation identifier is no longer part of a subscription identity.
MQPS_REGISTRATION_OPTIONS = MQPS_DUPLICATES_OK (MQCHAR*) or MQREGO_DUPLICATES_OK (MQLONG)	Duplicate publications are possible; see Overlapping topics and Grouping subscriptions.
MQPS_REGISTRATION_OPTIONS = MQPS_FULL_RESPONSE (MQCHAR*) or MQREGO_FULL_RESPONSE (MQLONG)	MQSUB returns all the response fields in the MQSD structure.
MQPS_REGISTRATION_OPTIONS = MQPS_INCLUDE_STREAM_NAME (MQCHAR*) or MQREGO_INCLUDE_STREAM_NAME (MQLONG)	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636
MQPS_REGISTRATION_OPTIONS = MQPS_INFORM_IF_RETAINED (MQCHAR*) or MQREGO_INFORM_IF_RETAINED (MQLONG)	The message property MQIsRetained, mqps.Ret, is returned with every publication.
MQPS_REGISTRATION_OPTIONS = MQPS_JOIN_EXCLUSIVE (MQCHAR*) or MQREGO_JOIN_EXCLUSIVE (MQLONG)	Subscriptions are not shared.
MQPS_REGISTRATION_OPTIONS = MQPS_JOIN_SHARED (MQCHAR*) or MQREGO_JOIN_SHARED (MQLONG)	Subscriptions are not shared.
MQPS_REGISTRATION_OPTIONS = MQPS_LOCAL (MQCHAR*) or MQREGO_LOCAL (MQLONG)	Set **MQPMO** option MQPMO_SCOPE_QMGR. **Note:** The meaning of **SCOPE** has changed; see "Publish/Subscribe: Local and global publications and subscriptions" on page 651
MQPS_REGISTRATION_OPTIONS = MQPS_LOCKED (MQCHAR*) or MQREGO_LOCKED (MQLONG)	Subscriptions are not shared.

Table 87. Migrate **Register Subscriber** *to MQSUB (continued)*

Register Subscriber	**MQSUB**
MQPS_REGISTRATION_OPTIONS = MQPS_NEW_PUBLICATIONS_ONLY(MQCHAR*) or MQREGO_NEW_PUBLICATIONS_ONLY (MQLONG)	Set the MQSUB option MQSO_NEW_ PUBLICATIONS_ONLY.
MQPS_REGISTRATION_OPTIONS = MQPS_NO_ALTERATION(MQCHAR*) or MQREGO_NO_ALTERATION (MQLONG)	Subscriptions are partly alterable; see Attributes that can be altered in MQSD and MQSUB.
MQPS_REGISTRATION_OPTIONS = MQPS_NON_PERSISTENT (MQCHAR*) or MQREGO_NON_PERSISTENT (MQLONG)	Set the MQSUB option MQSO_NON_PERSISTENT.
MQPS_REGISTRATION_OPTIONS = MQPS_PERSISTENT (MQCHAR*) or MQREGO_PERSISTENT (MQLONG)	Set the MQSUB option MQSO_PERSISTENT.
MQPS_REGISTRATION_OPTIONS = MQPS_PERSISTENT_AS_PUBLISH(MQCHAR*) or MQREGO_PERSISTENT_AS_PUBLISH (MQLONG)	Set the MQSUB option MQSO_PERSISTENT_AS_PUBLISH.
MQPS_REGISTRATION_OPTIONS = MQPS_PERSISTENT_AS_Q (MQCHAR*) or MQREFO_PERSISTENT_AS_Q (MQLONG)	Set the MQSUB option MQSO_PERSISTENT_AS_QUEUE_DEF.
MQPS_REGISTRATION_OPTIONS = MQPS_PUBLISH_ON_REQUEST_ONLY (MQCHAR*) or MQREGO_PUBLISH_ON_REQUEST_ONLY (MQLONG)	Set the MQSUB option MQSO_PUBLICATIONS_ON_REQUEST.
MQPS_REGISTRATION_OPTIONS = MQPS_VARIABLE_USER_ID (MQCHAR*) or MQREGO_VARIABLE_USER_ID (MQLONG)	Set the MQSUB option MQSO_ANY_USERID.
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636
MQPS_SUBSCRIPTION_NAME	The subscription is identified by the SubName field of the Subscription Descriptor, MQSD.
MQPS_SUBSCRIPTION_USER_DATA	Set the MQSD field, SubUserData.

Table 88. Migrate return codes

Reason code returned by the publish/subscribe broker	**Reason code returned by the queue manager**
MQRCCF_STREAM_ERROR	MQRC_HOBJ_ERROR
MQRCCF_TOPIC_ERROR	MQRC_HOBJ_ERROR
MQRCCF_Q_MGR_NAME_ERROR	MQRC_HOBJ_ERROR
MQRCCF_Q_NAME_ERROR	MQRC_HOBJ_ERROR
MQRCCF_DUPLICATE_IDENTITY	MQRC_IDENTITY_MISMATCH
MQRCCF_CORREL_ID_ERROR	
MQRCCF_NOT_AUTHORIZED	
MQRCCF_UNKNOWN_STREAM	MQRC_HOBJ_ERROR
MQRCCF_REG_OPTIONS_ERROR	
MQRCCF_DUPLICATE_SUBSCRIPTION	MQRC_SUB_ALREADY_EXISTS
MQRCCF_SUB_NAME_ERROR	
MQRCCF_SUB_IDENTITY_ERROR	Not applicable
MQRCCF_SUBSCRIPTION_IN_USE	MQRC_SUBSCRIPTION_IN_USE
MQRCCF_SUBSCRIPTION_LOCKED	Not applicable
MQRCCF_ALREADY_JOINED	Not applicable

Related information:

Registering as a subscriber

Register Subscriber

Register Subscriber (Format)

MQSD - Subscription descriptor

MQSUB

Publish/Subscribe Application migration: **Request Update** *replaced by MQSUBRQ:*

Replace the **Request Update** command by the MQSUBRQ Message Queue Interface (MQI) call. Migrate the **Request Update** parameters and options to the equivalent options in the MQSUBRQ calls.

Table 89. Migrate **Request Update** *to MQSUBRQ*

Request Update	**MQSUBRQ**
MQPS_COMMAND with value MQPS_REQUEST_UPDATE	Call MQSUBRQ setting the **Action** parameter option MQSR_ACTION_PUBLICATION. The request can be made only if the application set the option MQSO_PUBLICATIONS_ON_REQUEST on the MQSUB call when it made the subscription.
MQPS_TOPIC	The topic string is implied in the HSUB handle returned from the MQSUB call, which is used as a parameter on the MQSUBRQ call.
MQPS_Q_MGR_NAME	MQSUBRQ requires the connection handle of the queue manager to which the application is connected.
MQPS_Q_NAME	MQSUBRQ requires the HSUB subscription object handle returned by MQSUB.
MQPS_REGISTRATION_OPTIONS = MQPS_CORREL_ID_AS_IDENTITY (MQCHAR*) or MQREGO_CORREL_ID_AS_IDENTITY (MQLONG)	The subscriber identity is implicit in the connection handle obtained by MQSUB. It is released by MQCLOSE. The correlation identifier is no longer part of a subscription identity.
MQPS_REGISTRATION_OPTIONS = MQPS_VARIABLE_USER_ID (MQCHAR*) or MQREGO_VARIABLE_USER_ID (MQLONG)	Set the MQSUB option MQSO_ANY_USERID before obtaining the subscription handle.
MQPS_STREAM_NAME	See "Publish/Subscribe Application migration: Stream name replaced by topic object" on page 636
MQPS_SUBSCRIPTION_NAME	The subscription is identified by the SubName field of the Subscription Descriptor, MQSD.

Related information:

Request Update

Request Update (Format)

MQSUBRQ

Publish/Subscribe: Broker commands:

The commands used to control publish/subscribe have changed. The commands that have changed are listed.

Version 6.0 WebSphere MQ publish/subscribe broker is integrated into Version 7.5 publish/subscribe. The commands you used to control a Version 6.0 publish/subscribe broker are obsolete, and replaced by commands to control Version 7.5 publish/subscribe. The new commands, as they relate to controlling queued publish/subscribe, are described in Controlling queued publish/subscribe. Table 90 relates the old and new commands.

Table 90. WebSphere Message Broker command differences

Operation	IBM WebSphere MQ Version 6.0	IBM WebSphere MQ Version 7.5
Remove broker from hierarchy	clrmqbrk	See Disconnect a queue manager from a broker hierarchy for instructions how to disconnect a Version 7.5 queue manager from a hierarchy.
Delete broker	dltmqbrk	There is no publish/subscribe broker in Version 7.5. The **dltmqbrk** command removes Version 6.0 broker resources after running the command **strmqbrk** to migrate the Version 6.0 broker to Version 7.5 publish/subscribe.
Display broker	dspmqbrk	Use the **runmqsc** command DISPLAY PUBSUB to display publish/subscribe status.
Stop broker	endmqbrk	See Stopping queued publish/subscribe for instructions how to stop queued publish/subscribe in Version 7.5.
Migrate broker to WebSphere Message Broker	migmqbrk	Run the **migmqbrk** command on a Version 6.0 queue manager. Once you have upgraded, there is *no migration* from Version 7.5 publish/subscribe to the version 6.0 or 6.1 of the WebSphere Message Broker or WebSphere Event Broker publish/subscribe broker.
Start broker	strmqbrk	In Version 7.5, the **strmqbrk** command migrates publish/subscribe resources to Version 7.5. See Starting queued publish/subscribe for instructions how to start queued publish/subscribe in Version 7.5.

Publish/Subscribe: Adding a stream:

The method of adding a publish/subscribe stream to a queue manager has changed from Version 6.0. You can add stream to a queued publish/subscribe application. The new stream coexists with streams migrated from Version 6.0.

Related information:

Adding a stream
You can add streams manually so that they coexist with streams migrated from Version 6.0 queue managers.

Adding a stream, version 6.0

Publish/Subscribe: Connect a queue manager to a broker hierarchy:

The method of connecting a queue manager to a publish/subscribe hierarchy has changed from Version 6.0. You no longer define a connection with the **strmqbrk** command. A connection is defined by changing a queue manager attribute.

To connect a queue manager to a broker hierarchy, you must:

1. Start queued publish/subscribe for broker hierarchies to work.
2. Define the topology of the queue manager hierarchy by altering the parent the queue manager connects to.

You can combine a publish/subscribe broker running on Version 6.0, with a queue manager from a later version of IBM WebSphere MQ in the same hierarchy.

Example

```
ALTER QMGR PARENT(PARENTNAME)
```

Related information:

ALTER QMGR

Broker networks, version 6.0

Connect a queue manager to a broker hierarchy
You can connect a local queue manager to a parent queue manager to modify a broker hierarchy.

Starting queued publish/subscribe

strmqbrk, version 6.0

Publish/Subscribe: Deleting a stream:

The method of deleting a publish/subscribe stream from a queue manager has changed from Version 6.0. You can delete a stream that is no longer used by queue publish/subscribe applications.

Related information:

Deleting a stream
You can delete a stream from a IBM WebSphere MQ Version 7.1, or later, queue manager.

Deleting a stream, version 6.0

Publish/Subscribe: Disconnect a queue manager from a broker hierarchy:

The method of disconnecting a queue manager to a publish/subscribe hierarchy has changed from Version 6.0. You no longer disconnect a queue manager with the **dltmqbrk** command. A queue manager is disconnected from a hierarchy by changing a queue manager attribute.

In IBM WebSphere MQ Version 6.0, queue managers were disconnected from one another using the **dltmqbrk** command, and required that all child queue managers were disconnected first. The **dltmqbrk** command is now used to discard IBM WebSphere MQ Version 6.0 broker resources after migration using the **strmqbrk** command.

Disconnect a Version 7.5 queue manager from a broker hierarchy using the **ALTER QMGR** command. Unlike Version 6.0, you can disconnect queue managers in any order and at any time.

The corresponding request to update the parent is sent when the connection between the queue managers is running.

Example

```
ALTER QMGR PARENT('')
```

Related information:

ALTER QMGR

Broker networks, version 6.0

Disconnect a queue manager from a broker hierarchy
Disconnect a child queue manager from a parent queue manager in a broker hierarchy.

Publish/Subscribe: Queued publish/subscribe message attributes:

The behavior of some queued publish/subscribe message attributes is now controlled by queue manager attributes.

Set the following publish/subscribe attributes using IBM WebSphere MQ Explorer or the **runmqsc** command. In Version 6.0, the attributes in the publish/subscribe broker configuration stanza.

Table 91. Publish/subscribe attributes

Attribute	Description
PSRTYCNT	Command message retry count
PSNPMSG	Discard undeliverable command input message
PSNPRES	Behavior following undeliverable command response message
PSSYNCPT	Process command messages under syncpoint

Example

```
ALTER QMGR PSNPRES(SAFE)
```

Related information:

Queue manager parameters

Setting queued publish/subscribe message attributes
You control the behavior of some publish/subscribe message attributes using queue manager attributes. The other attributes you control in the *Broker* stanza of the `qm.ini` file.

⇨ Version 6 publish/subcribe broker configuration parameters

Publish/Subscribe: Starting queued publish/subscribe:

The task of starting the publish/subscribe broker has changed from running the **strmqbrk** command to enabling the queued publish/subscribe interface.

Enable queued publish/subscribe by setting the queue manager **PSMODE** attribute to `ENABLED`. The default is `ENABLED`.

Example

```
ALTER QMGR PSMODE(ENABLED)
```

Related information:

ALTER QMGR

PSMODE

Starting queued publish/subscribe

⇨ strmqbrk, version 6.0

Publish/Subscribe: Stopping queued publish/subscribe:

The task of stopping the publish/subscribe broker has changed from running the **endmqbrk** command to disabling the queued publish/subscribe interface.

Set the queue manager **PSMODE** attribute to `COMPAT` to stop the queued publish/subscribe interface. The integrated publish/subscribe interface continues to run. Setting the **PSMODE** attribute to `DISABLE` stops both the queued and integrated publish/subscribe interface.

In `COMPAT` mode, you can run integrated publish/subscribe alongside WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1 on the same server.

Example

```
ALTER QMGR PSMODE(COMPAT)
```

Related information:

ALTER QMGR

PSMODE

Stopping queued publish/subscribe

Publish/Subscribe: Configuration settings:

Configuration settings for the publish/subscribe broker in Version 6.0 are defined in the queue manager configuration file. Where equivalent settings exist, they are now queue manager attributes. The publish/subscribe migration program, **strmqbrk**, migrates the settings automatically.

Related information:

Broker configuration stanza

Working with queue managers

Publish/Subscribe: Delete temporary dynamic queue:

If a subscription is associated with a temporary dynamic queue, when the queue is deleted, the subscription is deleted. This changes the behavior of incorrectly written publish/subscribe applications migrated from Version 6.0. Publish/subscribe applications migrated from WebSphere Message Broker are unchanged. The change does not affect the behavior of integrated publish/subscribe applications, which are written using the MQI publish/subscribe interface.

- In IBM WebSphere MQ Version 6.0 the product documentation stated "The subscriber queue must not be a temporary dynamic queue"; see Queue name. But the injunction was not enforced, although the configuration is not supported.
- In WebSphere Message Broker and WebSphere Event Broker version 6.0, and WebSphere Message Broker version 6.1, you could create a subscription that used a temporary dynamic queue as the subscriber queue. If the queue was deleted, the subscription was deleted with it.
- In IBM WebSphere MQ Version 7.0, if you migrate or create a queued publish/subscribe application that uses `MQRFH1`, it behaves the same as IBM WebSphere MQ Version 6.0. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is not deleted, as in IBM WebSphere MQ Version 6.0. The lack of a subscriber queue results in any matching publications ending up on the dead letter queue. This behavior is the same as IBM WebSphere MQ Version 6.0. `MQRFH1` publish/subscribe applications are typically migrated from IBM WebSphere MQ Version 6.0.
- In IBM WebSphere MQ Version 7.0.1 from fix pack 7.0.1.6 onwards, and in Version 7.5, in the same `MQRFH1` queued publish/subscribe case, if the temporary dynamic queue is deleted, the subscription is deleted. This change prevents a buildup of publications from a subscription without a subscriber queue ending up on the dead letter queue. It is a change from IBM WebSphere MQ Version 6.0.
- In Version 7.5, if you migrate or create a queued publish/subscribe application that uses `MQRFH2`, it behaves the same as WebSphere Event Broker and WebSphere Message Broker. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is deleted, as in WebSphere Event Broker and WebSphere Message Broker. `MQRFH2` publish/subscribe applications are typically migrated from WebSphere Event Broker or WebSphere Message Broker.
- In IBM WebSphere MQ Version 7.5, if you create a durable subscription using integrated publish/subscribe, you cannot define a temporary dynamic queue as the destination for its matching publications.
- In IBM WebSphere MQ Version 7.5, you can create a managed, non-durable subscription using integrated publish/subscribe, which creates a temporary dynamic queue as the destination for matching publications. The subscription is deleted with the queue.

Summary

In Version 7.5, you cannot create a temporary dynamic queue as the destination for publications for a durable subscription using the integrated publish/subscribe interface.

In the current fix level of Version 7.5, if you use either of the queued publish/subscribe interfaces, `MQRFH1` or `MQRFH2`, the behavior is the same. You can create a temporary dynamic queue as the subscriber queue, and if the queue is deleted, the subscription is deleted with it. Deleting the subscription with the queue retains the same the supported behavior of IBM WebSphere MQ Version 6.0, WebSphere Event Broker, and WebSphere Message Broker applications. It modifies the unsupported behavior of IBM WebSphere MQ Version 6.0 applications.

Publish/Subscribe: Local and global publications and subscriptions:

The meaning of the local and global publication and subscription flags has changed from Version 6.0. In Version 6.0, local and global are attributes of the publication. Local or global are now attributes whether the subscription is propagated locally or globally.

The local or global publication scope is set by defining the scope of a subscription or a publication to `QMGR` or `ALL`. With this flag you can control, on a publication, whether it is propagated locally or globally. On a subscription, you can control whether it has an interest in publications made locally or globally. The flow of publications you observe is a result of the interplay of both these flags.

There has been a subtle change in the behavior between Version 6.0 and Version 7.5 if the publication and subscription flags are set to different options.

- In Version 7.5: if you specify that a subscription is only for local publications, the subscription receives any publications that are created on the local queue manager.
- In Version 6.0: if you set the same subscription option, you receive only publications that are created on this queue manager and are flagged for local publication only. Thus no publications are received if the publication scope is global, even if the publisher is local.
- In Version 7.5: if you specify that a publication is for global distribution, a subscription to the publication on any queue manager is eligible to receive the publication. Successful subscriptions include a subscription for local publications made on the same queue manager as the publisher.
- In Version 6.0: setting the same option, only subscribers that are eligible to receive global publications receive the publication. Thus no subscribers on the same queue manager as the publisher receive the publication, if their subscription has local scope.

The operation of publication and subscription scope in version 6.0 and Version 7.5 is described by Table 92 on page 652.

Note: If you specify opposite subscription scopes for the publisher and subscriber, it makes no difference which scope the publication or subscription is given. Either way round, in either version, the set of publications received is the same. In Version 6.0 no publications are received, and in Version 7.5 local publications are received.

Table 92. Version 6.0 and Version 7.5 publication and subscription scope truth table

Publisher **Subscriber**	**QMGR**		**ALL**	
	Version 6.0	**Version 7.5**	**Version 6.0**	**Version 7.5**
QMGR	No change: publications go to local subscribers		No publications are delivered	Only subscribers local to the publisher receive publications
ALL	No publications are delivered	Publications go to subscribers local to the publisher	No change: publications go to all subscribers	

Related information:

Controlling the flow of publications and subscriptions
Queue managers that are connected together into a distributed publish/subscribe topology share a common federated topic space. You can control the flow of publications and subscriptions within the topology by choosing whether each publication and subscription is either local or global.

Publish/Subscribe: Mapping an alias queue to a topic:

You can convert an existing application from a point-to-point to a publish/subscribe application, without changing the application. Change the application from referencing queues to referencing topics by defining or modifying an alias queue and associating it with a topic object. Existing applications are not affected unless they open an alias queue that you have associated with a topic.

Set the new QALIAS attribute **TARGTYPE** to TOPIC or QUEUE to specify that a queue alias resolves to a topic or a queue. The **TARGQ** attribute, defined in Version 6.0 as the name of the queue to which the alias queue resolves, is renamed to **TARGET**. The attribute name **TARGQ** is retained for compatibility with your existing programs. Set the name of the queue or topic using either **TARGET** or **TARGQ**.

Setting TARGTYPE(TOPIC) is useful to migrate existing applications to a publish/subscribe message model; see "Converting a monitoring application to use a subscription."

Queue-sharing groups and TARGTYPE

Within a queue sharing group it is possible to define a queue alias as a group object. Each queue manager in the queue sharing group creates a queue alias definition with the same name and the same properties as the QSGDISP(GROUP) object.

A queue manager can set or alter **TARGTYPE** in a QSGDISP(GROUP) alias queue object. Any Version 6.0 queue managers in the queue-sharing group ignore the **TARGTYPE** attribute. A Version 6.0 queue manager interprets the queue alias as a queue object, regardless of the setting of **TARGTYPE**.

Converting a monitoring application to use a subscription

A useful example of this feature is the queue to which statistics messages are written. In Version 6.0, only a single application could read a statistics message. Only a single statistics message was written to a queue and could be read from a queue.

By defining a queue alias that points to a topic object, each application that processes statistics messages can subscribe to a topic. Rather than a single application getting statistics messages from a queue, multiple applications can subscribe to and read the same statistics information.

Related information:

DEFINE QALIAS

DEFINE TOPIC

Working with alias queues

Publish/Subscribe: Message broker, or routing, exit changed to publish exit:

IBM WebSphere MQ Version 6.0 publish/subscribe has an exit for customizing and routing publications. The exit was renamed the publish exit in IBM WebSphere MQ Version 7.0.1. It was not available in IBM WebSphere MQ Version 7.0.

The message broker exit is largely replaced by using subscription levels. Using the **MQSD** `sublevel` field, an intermediate subscriber can intercept publications to customize or block them, before they arrive at the ultimate subscribers.

The function provided by intercepting publications using the **MQSD** `sublevel` field, and using the publish exit is similar. The main differences to be aware of are twofold:

1. If you use the subscription level mechanism, the intercepting application is a regular publish/subscribe application. You subscribe to the publications you intend to intercept. Publications are delivered to the intercepting application, which then must publish them again if they are to reach their intended destination.

   The Publish exit is written as an exit program and is called by the queue manager.
2. Intercept all publications from a publisher by using the subscription level mechanism. Run the intercepting application on the queue manager that the publisher is connected to.

   Intercept a publication just before it is delivered to a subscriber by using a publish exit. Configure the exit on the queue manager the subscriber is connected to.

Related information:

Intercepting publications

Message broker exit

Publish exit

Publish/Subscribe: Metatopics:

Metatopics are a special set of topics recognized by the IBM WebSphere MQ Version 6.0 publish/subscribe broker. Metatopics are no longer supported. Instead you can inquire on the list of topic names, and on individual topics, and subscriptions. If you send a subscription to a metatopic, the subscription is ignored.

Related information:

Metatopics

Publish/Subscribe: Multi-topic publications and subscriptions:

IBM WebSphere MQ Version 6.0 publish/subscribe has a concept of publishing to multiple topics, or subscribing to multiple topics, in a single publish or subscribe operation. IBM WebSphere MQ publish/subscribe no longer has these concepts. One consequence is that a Version 6.0 application running on a later version queue manager might receive two copies of a publication instead of one.

Version 6.0 publish/subscribe has separate concepts of a subscriber and a subscription. Publish/subscribe now has only the concept of a subscription. In Version 6.0, a single subscriber can have multiple subscriptions. If the same publication matches more than one subscription from the same subscriber, only one copy of the publication is received.

The closest publish/subscribe now comes to eliminating duplicate publications are the concepts of overlapping and grouped subscriptions. Setting the subscription option `MQSO_GROUP_SUB` eliminates duplicate instances of a publication that matches multiple subscriptions that are delivering publications to the same queue.

Version 6.0 also has the concept of a joint publication. A joint publication is a single publication to multiple topics. It behaves differently to separately publishing to each of the multiple topics in Version 6.0. Publish/subscribe no longer has a concept of a joint publication.

Example

Suppose a news service publishes the results of football matches. The service caters for fans of single or multiple teams who want to receive results for their favorite teams. For example, you might subscribe to the results for Chelsea and Manchester United. When the teams play together you want to receive just one result, but when the teams play in different matches you expect two results.

In Version 6.0, the news service publishes the results of each match as a joint result. When Manchester United and Chelsea play together the result, `Manchester United 1 :: Chelsea 0`, is published in a single publication to two topics, `/football/results/Manchester United` and `/football/results/Chelsea`. A football fan might subscribe, in Version 6.0, to all the teams they are following by registering a single subscription to multiple topics. A single Manchester United and Chelsea follower might register a subscription to both `/football/results/Manchester United` and `/football/results/Chelsea`.

With the Version 6.0 application running on a Version 6.0publish/subscribe broker, the fan receives a single publication when the teams play together. When they play separately, the fan receives separate publications for each match.

With the Version 6.0 application running on a later version queue manager the behavior is different. The single publish operation is broken down into two separate publish operations to the two different topics. The single subscription registration is broken down into two separate subscriptions, with the publications sent to the same queue. Consequently, when Manchester United and Chelsea play together, the fan receives two results.

The same problem arises if the application is split between a Version 6.0 and a later version queue manager. With either the publisher or subscriber running on the later version, the same difference arises.

In the example, `MQSO_GROUP_SUB` does not eliminate the duplication. The reason is two publications are created by emulating the joint publication. It is not a matter of a single publication matching multiple subscriptions.

Related information:

Grouping subscriptions

Overlapping subscriptions

⇨ Publish: Required parameters

⇨ Subscribe: Required parameters

Publish/Subscribe: Register Publisher and Deregister Publisher commands:

The Register Publish and Deregister Publisher commands do nothing any more, except return a successful response message to a request. Your publisher program is not affected by the change.

Related information:

Register Publisher

Deregister Publisher

Publish/Subscribe: Streams:

Streams are used in Version 6.0 to isolate publish/subscribe applications. Streams are now emulated. Migration from Version 6.0 streams is automatic. Existing Version 6.0 applications that use streams continue to work unchanged when migrated to, or interoperating with, later versions of IBM WebSphere MQ. In some unusual cases, you might need to modify the results of the automatic migration. If you introduce new streams to Version 6.0 applications, you must configure the emulated streams manually.

Version 6.0 topics can be isolated to a named stream, or topics can be created in the default stream, which is unnamed. Named streams can simplify the deployment of multiple publish/subscribe applications. If each application uses a different stream, there is no danger that publications for one application are received by subscriptions registered by a different application, even if the topic names are the same.

All topics now share the same topic space. Applications can still be isolated effectively, but the mechanism is different. Topic strings are split into two parts. The first part is managed administratively, and the second part is set by the application. An administrator creates topics that are used to isolate applications. These topics define the first part of a topic string. Applications define topics, or create topic strings dynamically, which then form the second part of a topics string. The complete topic string is constructed by an application combining the administrative topic with the application defined topic, or topic string. The application never has to refer to the topic string defined by the administrator, only to the topic name. The administrator can isolate, and modify, the topic spaces referenced by applications by creating, or modifying, the topics that define the first part of a topic string. These topics provide the same capability as streams do in Version 6.0.

Streams are emulated by mapping stream names to topics. The topics are added to the beginning of the topic string defined by an application. The mapping is controlled by a namelist, `SYSTEM.QPUBSUB.QUEUE.NAMELIST`, that identifies all the queues that are used by Version 6.0 streams that are now emulated by the queue manager. For details of the mechanism, and how to extend it to cater for unusual situations, see the related concepts.

Related information:

Adding a stream, version 6.0

Adding a stream
You can add streams manually so that they coexist with streams migrated from Version 6.0 queue managers.

Deleting a stream, version 6.0

Deleting a stream
You can delete a stream from a IBM WebSphere MQ Version 7.1, or later, queue manager.

Streams, version 6.0

Stream and topics
Queued publish/subscribe has the concept of a publication stream that does not exist in the integrated publish/subscribe model. In queued publish/subscribe, streams provide a way of separating the flow of information for different topics. In IBM WebSphere MQ Version 6.0, a stream is implemented as a queue, defined at each broker that supports the stream. Each queue has the same name (the name of the stream). From IBM WebSphere MQ Version 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

Publish/Subscribe: Subscription names:

In IBM WebSphere MQ Version 6.0, a subscription name is unique only within the stream and not across the queue manager. A subscription name must now be unique across the queue manager.

The **strqmbrk** command migrates Version 6.0 durable subscriptions. It ensures all subscriptions have names, and all the names are unique. If a subscription uses any other stream than `SYSTEM.BROKER.DEFAULT.STREAM`, the migration process appends the stream name to the subscription name. If the subscription uses `SYSTEM.BROKER.DEFAULT.STREAM` the subscription is migrated across without adding a stream name.

If the Version 6.0 subscription has not got a name, only a traditional identity, a subscription name is generated by **strqmbrk**. The name does not affect existing applications.

Related reference:

"JMS: Migration of subscriptions to named streams" on page 622

In Version 6.0 publish/subscribe, you can create subscriptions to topics in named streams, as well as to topics in the default, or unnamed, stream. Version 7.5 publish/subscribe does not have named streams. It emulates streams by modifying the topic string of a subscription that is associated with a stream. The result is, for most applications, the migration of subscription names from Version 6.0 to Version 7.5 is automatic. In the following cases, make changes to the results of the automatic migration.

Related information:

SubName (MQCHARV)

⇨ Subscription name and identity

Publish/Subscribe: Traditional identity:

The behavior of traditional identities differs between IBM WebSphere MQ Version 6.0 and later versions of IBM WebSphere MQ. In Version 6.0, subscriptions do not require names to identify them; a traditional identity is sufficient. In later versions of every subscription has a name. Subscription names are generated for any Version 6.0 subscriptions without a name that are migrated. Naming subscriptions has no affect on existing Version 6.0 applications. The relationship between traditional identities and subscriptions can result in different behavior.

Version 6.0 and later versions differ in how an identifier is assigned to a subscription. Version 6.0 identifies subscriptions in one or both of two ways. A combination of queue name, queue manager name, and optional correlation identifier always identifies a subscription in Version 6.0. This combination is called a traditional identity. In addition, you can add a subscription name to a traditional identity by associating a subscription name with it. You can then identify the subscription using the subscription name. JMS always identifies a subscription in Version 6.0 by using a subscription name.

In later versions, a subscription is identified only by a subscription name. When a Version 6.0 publish/subscribe broker is migrated by **strmqbrk**, subscription names are created automatically for any subscriptions that are not named. The behavior of Version 6.0 applications is not affected by the addition of a subscription name.

Note: Subscription names might change during the migration from Version 6.0; see the related reference.

A Version 6.0 application running on a later versions queue manager might receive more publications than it receives on Version 6.0. The difference is due to the way traditional identity behaves on Version 6.0. In Version 6.0, because of the coupling of subscription identity, and queue manager name, queue, and correlation identifier, the only publications delivered to a subscriber are subscriptions it registers. In later versions, a subscriber can create a subscription that delivers publications to any combination of queue manager, queue, and correlation identifier. The combination of queue manager, queue, and correlation identifier is not the identity of a subscription. As a result, if `MQSO_GROUP_SUB` is not set, a Version 6.0

subscriber might receive publications that it has not subscribed to.

Related reference:

"Publish/Subscribe: Subscription names" on page 656
In IBM WebSphere MQ Version 6.0, a subscription name is unique only within the stream and not across the queue manager. A subscription name must now be unique across the queue manager.

Related information:

DEFINE SUB

Subscription name and identity

Publish/Subscribe: Variable user ID:

In Version 6.0, only the user that creates a subscription is able to modify it, unless the user authorizes other users. The user does so by adding the subscription registration option, `MQPS_VARIABLE_USER_ID`, "`VariableUserId`". A similar rule now applies, but it is implemented differently. Only the user that creates a subscription is able to modify it, unless the subscription has the **`VARUSER`** attribute set to `ANY`. A publish/subscribe application can now also set the subscription option, `MQSO_ANY_USERID`, which has the same effect.

Existing Version 6.0 applications are unaffected by this difference. **`VARUSER`** has the same effect as using the Version 6.0 Register Subscriber command to set the `MQPS_VARIABLE_USER_ID`.

Note: You can modify a subscription created by a Version 6.0 application by setting the **`VARUSER`** attribute to `ANY`.

Related information:

MQSO_ANY_USERID subscription option

Register Subscriber optional parameters

Publish/Subscribe: Wildcards:

Publish/subscribe wildcard schemes have changed. Version 6.0 uses character-based wildcards. Now, you have a choice of using character-based and topic-based wildcards. The IBM WebSphere MQ topic-based wildcard scheme differs slightly from the topic-based wildcard scheme in WebSphere Event Broker version 6.0 and WebSphere Message Broker version 6.0 and 6.1. The IBM WebSphere MQ topic-based scheme allows multilevel wildcards in the middle of a topic string. The WebSphere Message Broker and WebSphere Event Broker scheme permits only topic multilevel wildcards at the end of a topic string.

The wildcard scheme defined by the MQ Telemetry Transport is the same as WebSphere Message Broker version 6.0.

The behavior of applications migrated from IBM WebSphere MQ Version 6.0 using queued publish/subscribe not affected. When writing new applications using integrated publish/subscribe interface, specify the subscription option, `MQSO_WILDCARD_CHAR`, to maintain compatibility with version 6.0 publish/subscribe programs.

Related information:

Wildcard schemes
There are two wildcard schemes used to subscribe to multiple topics. The choice of scheme is a subscription option.

Topic strings and topic filters in MQ Telemetry Transport clients

Wildcard characters

Queue manager logs: Default sizes increased

The default size of a queue manager log files has been changed to 4096. The `AMQERR`*`nn`*`.log` has increased from 256 KB to 2 MB on UNIX, Linux, and Windows platforms. The change affects both new and migrated queue managers.

Queue manager log

In IBM WebSphere MQ Version 7.5 default log size is 4096. For more information on setting non-default values, see The IBM WebSphere MQ configuration file, mqs.ini.

Queue manager error log

Override the change by setting the environment variable `MQMAXERRORLOGSIZE`, or setting `ErrorLogSize` in the `QMErrorLog` stanza in the `qm.ini` file.

The change increases the number of error messages that are saved in the error logs.

Related information:

Queue manager error logs
Use the `Extended` queue manager properties page from the WebSphere MQ Explorer, or the `QMErrorLog` stanza in the qm.ini file to tailor the operation and contents of queue manager error logs.

SSL or TLS Authority Information Access (AIA) checked by default

If a TLS or SSL certificate has an AIA extension, the certificate is checked by default. If the certificate contains a URL, the WebSphere MQ channel tries to contact the Online Certificate Status Protocol (OCSP) responder at the URL. As a result channels might take a long time to start, or not start, depending on the delay in receiving a response. You can set up an OCSP proxy server to overcome firewall restrictions, or turn off AIA checking to improve performance.

The URL contained in the AIA extension often cannot be reached because of firewall restrictions. The typical solution to accessing an OCSP responder through a firewall is to set up an OCSP proxy server. WebSphere MQ supports proxy servers by providing the `SSLHTTPProxyName` configuration file variable, or on a client, the `MQSSLPROXY` environment variable.

The timeout on the attempt to contact the OCSP responder can be long. The response delay on contacting the OCSP can cause channels to take a long time to start. Channels might timeout, if they are running with short heartbeat intervals.

You can turn off AIA certificate extension checking using the `OCSPCheckExtensions` configuration file variable. You might safely turn off SSL/TLS certificate revocation checking in a test environment. In a production environment, if a certificate has an AIA extension it is important that the certificate is checked with the OCSP responder. If you do not check whether a certificate is revoked, a user presenting a revoked certificate would get unauthorized access to your system.

Related information:

Online Certificate Status Protocol - OCSP

Revoked certificates and OCSP

SSL stanza of the client configuration file
Use the SSL stanza to specify information about the use of SSL or TLS.

SSL stanza of the queue manager configuration file
Use the SSL stanza of the queue manager configuration file to configure SSL or TLS channels on your queue manager.

SSLPEER and SSLCERTI changes

IBM WebSphere MQ Version 7.5 obtains the Distinguished Encoding Rules (DER) encoding of the certificate and uses it to determine the subject and issuer distinguished names. The subject and issuer distinguished names are used in the SSLPEER and SSLCERTI fields. A SERIALNUMBER attribute is also included in the subject distinguished name and contains the serial number for the certificate of the remote partner. Some attributes of subject and issuer distinguished names are returned in a different sequence from previous releases.

The change to subject and issuer distinguished names affects channel security exits. It also affects aplications which depend upon the subject and issuer distinguished names that are returned by the PCF programming interface. Channel security exits and applications that set or query SSLPEER and SSLCERTI must be examined, and possibly changed. The fields that are affected are listed in Table 93 and Table 94.

Table 93. Channel status fields affected by changes to subject and issuer distinguished names

Channel status attribute	PCF channel parameter type
SSL Peer (SSLPEER)	MQCACH_SSL_SHORT_PEER_NAME
SSLCERTI	MQCACH_SSL_CERT_ISSUER_NAME

Table 94. Channel data structures affected by changes to subject and issuer distinguished names

Channel data structure	Field
MQCD - Channel definition	SSLPeerNamePtr (MQPTR)
MQCXP - Channel exit parameter	SSLRemCertIssNamePtr (PMQVOID)

Existing peer name filters specified in the SSLPEER field of a channel definition are not affected. They continue to operate in the same manner as in earlier releases. The peer name matching algorithm has been updated to process existing SSLPEER filters. It is not necessary to alter any channel definitions.

Related information:

Channel security exit programs

Preferred alternatives: Consider replacing these Version 6.0 functions with their Version 7.5 alternatives

Some functions in version 6.0 are replaced by alternatives in Version 7.5. The Version 6.0 functions continue to work in Version 7.5 for compatibility reasons. Write new applications using the Version 7.5 alternatives. In some cases, new Version 7.5 function is not available to an application, unless you migrate it to use the Version 7.5 alternative.

Table 95. List of functions in Version 6.0 that have preferred Version 7.5 alternatives

Version 6.0	Version 7.5	Description
Queued Publish/Subscribe	Integrated Publish/Subscribe	"Publish/Subscribe: Application migration" on page 634
JMS classes		Deprecated JMS API
Java classes		Deprecated Java API
com.ibm.mq.MQC	`com.ibm.mq.constants.MQConstants`	A new package, `com.ibm.mq.constants`, is supplied with WebSphere MQ Version 7.5. `com.ibm.mq.constants` package contains the class `MQConstants`, which implements a number of interfaces. `MQConstants` contains definitions of all the constants that were in the MQC interface and a number of new constants. The interfaces in this package closely follow the names of the constants header files used in WebSphere MQ. For example, the interface `CMQC` contains a constant `MQOO_INPUT_SHARED`; this interface corresponds to the header file `cmqc.h` and the constant `MQOO_INPUT_SHARED`. `com.ibm.mq.constants` can be used with both WebSphere MQ classes for Java and WebSphere MQ classes for JMS. MQC is still present, and has the constants it previously had; however, for any new applications, you must use the `com.ibm.mq.constants` package.
Java System property `com.ibm.mq.exitClasspath`	Set `JavaExitsClasspath` in `mqclient.ini`	See ClientExitPath stanza of the client configuration file
STRMQMCHLI	Set up a transmission queue specifying `SYSTEM.CHANNEL.INITQ` as the initiation queue and enabling triggering.	See Triggering channels in WebSphere MQ for IBM i

IBM WebSphere MQ Explorer changes

IBM WebSphere Eclipse Platform is no longer shipped with IBM WebSphere MQ; it is not required to run MQ Explorer. The change makes no difference to administrators who run MQ Explorer. For developers who run MQ Explorer in an Eclipse development environment, a change is necessary. You must install and configure a separate Eclipse environment to be able to switch between MQ Explorer and other perspectives.

Packaging changes

In versions of IBM WebSphere MQ earlier than Version 7.5, you can select the Workbench mode preference in MQ Explorer. In workbench mode, you could switch to the other perspectives installed in the WebSphere Eclipse Platform. You can no longer set the Workbench mode preference, because the WebSphere Eclipse Platform is not shipped with MQ Explorer in Version 7.5.

To switch between MQ Explorer and other perspectives, you must install MQ Explorer into your own Eclipse environment or into an Eclipse-based product. You can then switch between perspectives. For example, you can develop applications using IBM WebSphere MQ classes for JMS or IBM WebSphere MQ Telemetry applications; see Creating your first MQ Telemetry Transport publisher application using Java

If you installed extensions to previous versions of MQ Explorer, such as SupportPacs or WebSphere Message Broker Explorer, you must reinstall compatible versions of the extensions after upgrading MQ Explorer to Version 7.5.

If you continue to run IBM WebSphere MQ Version 7.0.1 on the same server as IBM WebSphere MQ Version 7.5, and you use MQ Explorer, each installation uses its own installation of MQ Explorer. When you uninstall Version 7.0.1, its version of MQ Explorer is uninstalled. To remove IBM WebSphere Eclipse Platform, uninstall it separately. The workspace is not deleted.

Authorization services preference migration

The way that the IBM WebSphere MQ Explorer authorization services preferences are stored has changed. Therefore, when migrating to Version 7.5, the preferences are not migrated and must be manually set.

SSL client certificate store migration

After migrating SSL client certificate stores to Version 7.5, you must select the **Enable default SSL key repositories** check box to use the SSL client certificate stores.

Test result migration

Test results are not migrated from version to version. To view any test results, you must rerun the tests.

More information

For more information, see the WebSphere MQ Explorer documentation at WebSphere MQ Explorer.

AIX: Shared objects

In IBM WebSphere MQ for AIX Version 7.5 the `.a` shared objects in the `lib64` directory contains both the 32 bit and 64 bit objects. A symlink to the `.a` file is also placed in the `lib` directory. The AIX loader can then correctly pick up the correct object for the type of application being run.

This means that IBM WebSphere MQ applications can run with the `LIBPATH` containing either the `lib` or `lib64` directory, or both.

AIX: `/usr/lpp/mqm` symbolic link removed

Before Version 6.0, IBM WebSphere MQ placed a symbolic link in `/usr/lpp/mqm` on AIX. The link ensured queue managers and applications migrated from IBM WebSphere MQ versions before Version 5.3 continued to work, without change. The link is not created in Version 7.5.

In version 5.0, IBM WebSphere MQ for AIX was installed into `/usr/lpp/mqm`. That changed in Version 5.3 to `/usr/mqm`. A symbolic link was placed in `/usr/lpp/mqm`, linking to `/usr/mqm`. Existing programs and scripts that relied on the installation into `/usr/lpp/mqm` continued to work unchanged. That symbolic link has been removed in Version 7.5, because you can now install IBM WebSphere MQ in any directory. Applications and command scripts are affected by the change.

The effect on applications is no different to the effect of migrating on other UNIX and Linux platforms. If the installation is made primary, then symbolic links to the IBM WebSphere MQ link libraries are placed in `/usr/lib`. Most applications migrated from earlier IBM WebSphere MQ versions search the default search path, which normally includes `/usr/lib`. The applications find the symbolic link to the IBM WebSphere MQ load libraries in `/usr/lib`.

If the installation is not primary, then you must configure the correct search path to load the IBM WebSphere MQ link libraries. If you choose to run **`setmqenv`**, IBM WebSphere MQ places the IBM WebSphere MQ link library path into `LIBPATH`. Unless the application is configured not to search the

LIBPATH, if for example it is a setuid or setgid application, then the IBM WebSphere MQ library is loaded successfully; see "UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488.

If you have written command scripts that run IBM WebSphere MQ commands, you might have coded explicit paths to the directory tree where IBM WebSphere MQ was installed. You must modify these command scripts. You can run **setmqenv** to create the correct environment to run the command scripts. If you have set the installation as primary, you do not have to specify the path to the command.

Related tasks:

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Related information:

"Loading IBM WebSphere MQ libraries" on page 350

AIX, HP-UX, and Solaris: Building applications for TXSeries

You must rebuild IBM WebSphere MQ applications that link to TXSeries.

Before Version 7.5, IBM WebSphere MQ applications that used the TXSeries CICS support loaded the IBM WebSphere MQ library, mqz_r. From Version 7.5 those applications must load the IBM WebSphere MQ library, mqzi_r instead. You must change your build scripts accordingly.

Version 7.5 of mqz_r includes code to load a different version of the IBM WebSphere MQ library. IBM WebSphere MQ loads a different version of the IBM WebSphere MQ library, if it detects that the queue manager the application is connected to is associated with a different installation to the one from which the library was loaded. mqzi_r does not include the additional code. When using TXSeries, the application must run with the IBM WebSphere MQ library it loaded, and not a different library loaded by IBM WebSphere MQ. For this reason, IBM WebSphere MQ applications that use the IBM WebSphere MQ TXSeries support must load the mqzi_r library, and not the mqz_r library.

An implication of applications loading mqzi_r is that the application must load the correct version of mqzi_r. It must load the one from the installation that is associated with the queue manager that the application is connected to.

Related information:

Building libraries for use with TXSeries for Multiplatforms version 5

AIX: TXSeries CICS support

HP-UX: TXSeries CICS support

Solaris: TXSeries CICS support

Linux: Recompile C++ applications and update run time libraries

C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in /usr/lib or /usr/lib64

If you are using one of the supported Linux distributions, the libraries are correctly installed; see IBM WebSphere MQ requirements.

The GCC 4.1.2 libraries support SSL and TLS connections from a IBM WebSphere MQ MQI client. SSL and TLS use GSKit version 8, which depends on libstdc++.so.6. libstdc++.so.6 is included in GCC 4.1.2.

Related tasks:

"Linux: Rebuilding a C++ application" on page 491
C++ IBM WebSphere MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

UNIX and Linux: `crtmqlnk` and `dltmqlnk` removed

The **`crtmqlnk`** and **`dltmqlnk`** commands are not present in this version of the product. In versions prior to Version 7.1, the commands created symbolic links in subdirectories of `/usr`. From Version 7.1 onwards, you must use the **`setmqinst`** command instead.

Related reference:

"UNIX and Linux: `/usr` symbolic links removed" on page 591
On all UNIX and Linux platforms, the links from the `/usr` file system are no longer made automatically. In order to take advantage of these links, you must set an installation as the primary installation or set the links up manually.

Related information:

Changing the primary installation
You can use the **`setmqinst`** command to set or unset an installation as the primary installation.

setmqinst

UNIX and Linux: Message catalogs moved

IBM WebSphere MQ message catalogs are no longer stored in the system directories in Version 7.5. To support multiple installations, copies of the message catalogs are stored with each installation. If you want messages only in the locale of your system, the change has no affect on your system. If you have customized the way the search procedure selects a message catalog, then the customization might no longer work correctly.

Set the `LANG` environment variable to load a message catalog for a different language from the system locale.

Related information:

Displaying messages in your national language on UNIX and Linux systems
To display messages from a different national language message catalog, you must install the appropriate catalog and set the **`LANG`** environment variable.

UNIX and Linux: MQ services and triggered applications

IBM WebSphere MQ Version 7.5 has been configured to allow both `LD_LIBRARY_PATH` and `$ORIGIN` to work for MQ services and triggered applications. For this reason MQ Services and triggered applications have been changed so that they run under the user ID who started the queue manager and not `setuid` or `setgid`.

If any files used by the service were previously restricted to certain users, then they might not be accessible by the user ID who started the queue manager. Resources used by MQ services or triggered applications must be adjusted as appropriate.

Note:

- On AIX, `LD_LIBRARY_PATH` is also known as `LIBPATH` and `$ORIGIN` is not supported.
- On HP-UX, `LD_LIBRARY_PATH` is also known as `SHLIB_PATH`.

Related information:

Working with services

Starting WebSphere MQ applications using triggers

UNIX and Linux: `ps -ef | grep amq` interpretation

The interpretation of the list of IBM WebSphere MQ processes that results from filtering a scan of UNIXor Linux processes has changed. The results can show IBM WebSphere MQ processes running for multiple installations on a server. Before Version 7.5, the search identified IBM WebSphere MQ processes running on only a single installation of IBM WebSphere MQ on a UNIXor Linux server.

The implications of this change depend on how the results are qualified and interpreted, and how the list of processes is used. The change affects you, only if you start to run multiple installations on a single server. If you have incorporated the list of IBM WebSphere MQ processes into administrative scripts or manual procedures, you must review the usage.

Examples

The following two examples, which are drawn from the product documentation, illustrate the point.

1. In the product documentation, before Version 7.5, the scan was used as a step in tasks to change the installation of IBM WebSphere MQ. The purpose was to detect when all queue managers had ended. In Version 7.5, the tasks use the **dspmq** command to detect when all queue managers associated with a specific installation have ended.
2. In the product documentation, a process scan is used to monitor starting a queue manager in a high availability cluster. Another script is used to stop a queue manager. In the script to stop a queue manager, if the queue manager does not end within a period of time, the list of processes is piped into a **kill -9** command. In both these cases, the scan filters on the queue manager name, and is unaffected by the change to multiple installations.

Related information:

Stopping a queue manager under the control of a high availability (HA) cluster
In most cases, you can use a shell script to stop a queue manager. Examples of suitable shell scripts are given here. You can tailor these to your needs and use them to stop the queue manager under control of your HA cluster.

Monitoring a queue manager
It is usual to provide a way for the high availability (HA) cluster to monitor the state of the queue manager periodically. In most cases, you can use a shell script for this. Examples of suitable shell scripts are given here. You can tailor these scripts to your needs and use them to make additional monitoring checks specific to your environment.

UNIX and Linux: `/usr` symbolic links removed

On all UNIX and Linux platforms, the links from the `/usr` file system are no longer made automatically. In order to take advantage of these links, you must set an installation as the primary installation or set the links up manually.

In previous releases the installation of IBM WebSphere MQ on UNIX and Linux created the symbolic links shown in Table 96 on page 665. In Version 7.5, these links are not created. You must run **setmqinst** to create a primary installation containing symbolic links. No symbolic links are created in other installations.

Table 96. Default symbolic links created in releases before Version 7.1

Symbolic link from	To
/usr/bin/amq...	/opt/mqm/bin/amq...
/usr/lib/amq...	/opt/mqm/lib/amq...
/usr/include/cmq...	/opt/mqm/inc/cmq...
/usr/share/man/...	/opt/mqm/man/...

Only a subset of those links created with previous releases are now made, see External library and control command links to primary installation on UNIX and Linux

For more information about choosing whether to have a primary installation, see Choosing a primary installation

Related tasks:

"UNIX and Linux: Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 488
Investigate whether applications connecting to IBM WebSphere MQ for Windows Version 7.5 are linked to, and load libraries from, the correct installation.

Related information:

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

setmqinst

Windows: `amqmsrvn.exe` process removed

The `amqmsrvn.exe` DCOM process was replaced by a Windows service, `amqsvc.exe`, in Version 7.1. This change is unlikely to cause any problems. However, you might have to make some changes. You might have configured the user that runs the IBM WebSphere MQ Windows service `MQSeriesServices` without the user right to "Log on as a service". Alternatively, the user might not have "List Folder" privilege on all the subdirectories from the root of the drive to the location of the service `amqsvc.exe`.

If you omitted the "Log on as a service" user privilege, or one of the subdirectories under which IBM WebSphere MQ is installed does not grant the "List Folder" privilege to the user, the MQ_*InstallationName* IBM WebSphere MQ Windows services in Version 7.5 fails to start.

Diagnosing the problem

If the service fails to start, Windows event messages are generated:

- If you did not give the user the "Log on as a service" user privilege, the Windows Service Control Manager adds an event: `7038: The user has not been granted the requested logon type`. The **strmqsvc** command reports error 1069.
- If you did not give the user the "List Folder" privilege, the Windows Service Control Manager adds an event: `7009: Timed out waiting for the service to connect`. The **strmqsvc** command reports error 1053.

If the "Prepare WebSphere MQ" wizard encounters a failure when validating the security credentials of the user performing an installation, an error is returned: `WebSphere MQ is not correctly configured for Windows domain users`. This error indicates that the service failed to start.

Resolution

To resolve this problem:

- Check that the user has the "Log on as a service" user privilege
- Check that the user is a member of the local mqm group

- Check that the local mqm group has “List Folder” privilege on each subdirectory in the path to the service amqsvc.exe.

Related reference:

“Windows: “Logon as a service” required” on page 594
The user ID that runs the IBM WebSphere MQ Windows service must have the user authority to “Logon as a service”. If the user ID does not have the authority to run the service, the service does not start and it returns an error in the Windows system event log. Typically you will have run the Prepare IBM WebSphere MQ wizard, and set up the user ID correctly. Only if you have configured the user ID manually is it possible that you might have a problem in Version 7.5.

Windows: IgnoredErrorCodes registry key

The registry key used to specify error codes that you do not want written to the Windows Application Event Log has changed.

The contents of this registry key are not automatically migrated. If you want to continue to ignore specific error codes, you must manually migrate the registry key.

Previously, the key was in the following location:

```
HKLM\Software\IBM\MQSeries\CurrentVersion\IgnoredErrorCodes
```

The key is now in the following location:

```
HKLM\Software\IBM\WebSphere MQ\Installation\MQ_INSTALLATION_NAME\IgnoredErrorCodes
```

where *MQ_INSTALLATION_NAME* is the installation name associated with a particular installation of IBM WebSphere MQ.

Related information:

Ignoring error codes under Windows systems

Windows: Installation and infrastructure information

The location of Windows installation and infrastructure information has changed.

A top-level string value, WorkPath, in the HKLM\SOFTWARE\IBM\WebSphere MQ key, stores the location of the product data directory which is shared between all installations. The first installation on a machine specifies it, subsequent installations pick up the same location from this key.

Other information previously stored in the registry on Windows is now stored in .ini files.

Related information:

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Windows: Local queue performance monitoring

In IBM WebSphere MQ for Windows Version 7.5 it is no longer possible to monitor local queues using the Windows performance monitor.

Use the performance monitoring commands, which are common to all platforms, provided by IBM WebSphere MQ.

Related information:

Real-time monitoring

Windows: "Logon as a service" required

The user ID that runs the IBM WebSphere MQ Windows service must have the user authority to "Logon as a service". If the user ID does not have the authority to run the service, the service does not start and it returns an error in the Windows system event log. Typically you will have run the Prepare IBM WebSphere MQ wizard, and set up the user ID correctly. Only if you have configured the user ID manually is it possible that you might have a problem in Version 7.5.

You have always been required to give the user ID that you configure to run IBM WebSphere MQ the user authority to "Logon as a service". If you run the Prepare IBM WebSphere MQ wizard, it creates a user ID with this authority. Alternatively, it ensures that a user ID you provide has this authority.

It is possible that you ran IBM WebSphere MQ in earlier releases with a user ID that did not have the "Logon as a service" authority. You might have used it to configure the IBM WebSphere MQ Windows service MQSeriesServices, without any problems. If you run a IBM WebSphere MQ Windows service in Version 7.5 with the same user ID that does not have the "Logon as a service" authority, the service does not start.

The IBM WebSphere MQ Windows service MQSeriesServices, with the display name `IBM MQSeries`, changed in Version 7.1. A single IBM WebSphere MQ Windows service per server is no longer sufficient. A IBM WebSphere MQ Windows service per installation is required. Each service is named `MQ_`*`InstallationName`*, and has a display name `IBM WebSphere MQ (`*`InstallationName`*`)`. The change, which is necessary to run multiple installations of IBM WebSphere MQ, has prevented IBM WebSphere MQ running the service under a single specific user ID. In Version 7.5, a `MQ_`*`InstallationName`* service must run as a service.

The consequence is a user ID that is configured to run the Windows service `MQ_`*`InstallationName`* must be configured to "Logon as a service". If the user ID is not configured correctly, errors are returned in the Windows system event log.

Many installations on earlier releases, and installations from Version 7.1 onwards, configure IBM WebSphere MQ with the Prepare IBM WebSphere MQ wizard. The wizard sets up the user ID with the "Logon as a service" authority and configures the IBM WebSphere MQ Windows service with this user ID. Only if, in previous releases, you have configured MQSeriesServices with another user ID that you configured manually, might you have this migration problem to fix.

Windows: Migration of registry information

Before Version 7.1 all IBM WebSphere MQ configuration information, and most queue manager configuration information, was stored in the Windows registry. From Version 7.1 onwards, all queue manager configuration information (for example, `mqs.ini`, `qmstatus.ini`, and `qm.ini`) is stored in files; the same files as in UNIX and Linux platforms.

The change does not affect the operation of existing applications or queue managers, but it does affect any administrative procedures and scripts that reference the registry.

If you are migrating an existing Windows system to version 7.5, the transfer of configuration data from the registry to files is automatic. It takes place when the installation is upgraded to version 7.5.

Related information:

Changing configuration information on Windows, UNIX and Linux systems
Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

Windows: MSCS restriction with multiple installations

When you install or upgrade to IBM WebSphere MQ Version 7.5, the first IBM WebSphere MQ installation on the server is the only one that can be used with Microsoft Cluster Server (MSCS). No other installations on the server can be used with MSCS. This restriction limits the use of MSCS with multiple IBM WebSphere MQ installations.

When you run the **haregtyp** command it defines the first IBM WebSphere MQ to be installed as an MSCS resource type; see WebSphere MQ MSCS support utility programs. The implications are as follows:

1. You must associate queue managers that are participating in an MSCS cluster with the first installation on the server.
2. Setting the primary installation has no effect on which installation is associated with the MSCS cluster.

Windows: Relocation of the mqclient.ini file

In IBM WebSphere MQ for Windows Version 7.5 the `mqclient.ini` file has moved from FilePath to WorkPath. This is similar to the model already used on UNIX and Linux systems.

For users who supply separate file and work paths you will see a change in behavior. You have an additional step to perform when you choose to uninstall IBM WebSphere MQ Version 7.0 before installing IBM WebSphere MQ Version 7.5. Before uninstalling IBM WebSphere MQ Version 7.0, you must copy `mqclient.ini` directly to the Config directory in your data path so that it can be picked up by the IBM WebSphere MQ Version 7.5 installation.

Windows: SPX support on Windows Vista

The Sequenced Package Exchange protocol (SPX) is not supported on Windows Vista or Windows Server 2008. SPX is supported on Windows XP and Windows 2003 only.

Related information:

Defining an SPX connection
An SPX connection applies only to a client and server running Windows XP and Windows 2003 Server.

Windows: Task manager interpretation

The interpretation of the processes listed by the Windows Task Manager has changed. The results can show IBM WebSphere MQ processes running for multiple installations on a server. Before Version 7.5, the process list identified IBM WebSphere MQ processes running on only a single installation of IBM WebSphere MQ on a Windows server.

The implications of this change depend on how the results are qualified and interpreted, and how the list of processes is used. The change affects you, only if you start to run multiple installations on a single server. If you have incorporated the list of IBM WebSphere MQ processes into administrative scripts or manual procedures, you must review the usage.

Related information:

Stopping queue managers in WebSphere MQ for Windows

Windows: WebSphere MQ Installation affected by User Account Control on Windows systems

On some Windows systems, installation of WebSphere MQ is interrupted at various points by User Account Control (UAC). You must acknowledge the Windows UAC prompt to allow installation to run with elevated authority. You must run silent installation from an elevated command prompt.

UAC is enabled by default. UAC restricts the actions users can perform on certain operating system facilities, even if they are members of the Administrators group.

At certain points during interactive installation, migration, and uninstallation, you must acknowledge the Windows UAC prompt to allow installation to run with elevated authority. The points at which you must accept the Windows prompt for UAC are noted in the specific topics affected.

If you opt to run silent installation, you must start the installation process from an elevated command prompt. The points at which you must use an elevated command prompt are noted in the specific topics affected.

Related information:

Advanced installation using msiexec

Windows: IBM WebSphere MQ Active Directory Services Interface

The IBM WebSphere MQ Active Directory Services Interface is no longer available.

If your application uses the IBM WebSphere MQ Active Directory Services Interface, you must rewrite your application to use Programmable Command Formats.

Related information:

Introduction to Programmable Command Formats

Differences between the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker version 6, and the publish/subscribe broker in WebSphere MQ version 7

WebSphere Event Broker version 6 is superseded by the publish/subscribe broker in WebSphere MQ version 7.0.1. The internal publish/subscribe broker in WebSphere Message Broker version 6.0 and 6.1 is replaced by WebSphere MQ version 7.0.1 publish/subscribe in WebSphere Message Broker version 7. Applications are affected by a number of changes in behavior by the migration to the publish/subscribe broker in WebSphere MQ.

Related concepts:

"Migration of the publish/subscribe broker in WebSphere Event Broker and WebSphere Message Broker" on page 369
The publish/subscribe broker used in WebSphere Message Broker version 6.0 and 6.1 is replaced by the integrated publish/subscribe broker in IBM WebSphere MQ. IBM WebSphere MQ replaces WebSphere Event Broker version 6.0. The migration to IBM WebSphere MQ has two parts. The migration of publish/subscribe configuration data, and conversion of publish/subscribe collectives to publish/subscribe clusters.

Related tasks:

"WebSphere Event Broker and WebSphere Message Broker migration tasks" on page 546
Migrate the publish/subscribe broker and collectives from WebSphere Event Broker or WebSphere Message Broker version 6.0 or 6.1 to Version 7.5. Migrate SCADA applications that use IBM WebSphere MQ Telemetry.

Related information:

Migrating publish/subscribe information to WebSphere MQ Version 7.0.1

Publish/Subscribe Broker: ACL migration on Linux and UNIX platforms

On Linux and UNIX platforms, it is possible for a username and groupname to have the same identifier. In which case the migration process is unable to determine whether an ACL applies to the username, the groupname, or both.

The ACL migration script contains two **setmqaut** commands, one of which is commented out. The first command applies to the username and the second command applies to the group name. Here is an example of the commands:

```
setmqaut -m QM1 -n SAMPLE_TOPIC_OBJECT -t topic -p mquser +sub +pub
# setmqaut -m QM1 -n SAMPLE_TOPIC_OBJECT -t topic -g mquser +sub +pub
```

The second command is prefixed by a number sign "#" to indicate that it is commented out. As part of reviewing the ACL migration script you must ensure that the correct command is executed.

Related concepts:

"Publish/Subscribe Broker: Access Control List (ACL) migration" on page 371
An automated process migrates the publish/subscribe broker from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ. The process does not migrate the ACLs automatically. The migration task provides two methods to migrate the ACL to WebSphere MQ.

Publish/Subscribe Broker: ACL wildcard subscriptions

A subscription can use a wildcard scheme to subscribe to multiple topics. Access to one or more of the resulting matched topics might be denied by its access control list. The result in WebSphere MQ differs after migration from the result in WebSphere Event Broker and WebSphere Message Broker before migration.

- In the brokers, the subscription succeeds, but the brokers do not publish messages to topics for which access authority is denied. For example, ACLs are set up in the brokers that grant access to the topic `USA/Alaska`, and deny access to the topic `USA/Kansas`. In the brokers, a subscription to `USA/#` or `USA/+` succeeds, but the subscription does not receive messages that are published to USA/Kansas.
- In WebSphere MQ, for a subscription to succeed, it must have access authority to all of the topics to which the wildcard scheme subscribes. Therefore the subscription fails. For example, access permissions are set up in a queue manager that grant access to the topic `USA/Alaska` and deny access to the topic `USA/Kansas`. A subscription to `USA/#` or `USA/+` fails. Because the subscription fails, the subscription does not receive messages that are published to either `USA/Alaska` or `USA/Kansas`.

As a result, even though you have set up an equivalent security environment in the queue manager, some subscriptions might fail to be migrated. The solution to the wildcard problem is not straightforward, because of the difference in behavior between the two products. You cannot solve the problem by changing access permissions alone. You either must accept different results, or create different subscriptions that you can secure to give the same results. If you can accept that a subscriber to `USA/#` or `USA/+` is to receive messages published to `USA/Kansas`, you must grant permission to the topic `USA/Kansas`. If you cannot accept that a subscriber to `USA/#` or `USA/+` is to receive messages published to `USA/Kansas` you must change the subscription. One solution is to replace the wildcard subscription with multiple subscriptions lower down the topic tree.

Related concepts:

"Publish/Subscribe Broker: Access Control List (ACL) migration" on page 371
An automated process migrates the publish/subscribe broker from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ. The process does not migrate the ACLs automatically. The migration task provides two methods to migrate the ACL to WebSphere MQ.

Publish/Subscribe Broker: Adding a subscription point

The method of adding a subscription point has changed from WebSphere Event Broker and WebSphere Message Broker version 6.0 and version 6.1. You must add the subscription point to IBM WebSphere MQ and to message flows in the broker. Existing subscription points are migrated by the **`migmbbrk`** command.

Related information:

Adding a subscription point
How to add a subscription point that was not migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker by **`migmbbrk`**. Extend an existing queued publish/subscribe application that you have migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker with a new subscription point.

Subscription points

Publish/Subscribe Broker: Altering fields

In WebSphere Message Broker version 6.0 or 6.1, and WebSphere Event Broker version 6, you can alter the topic, filter, destination queue and subscription name fields of a subscription. In version 7.0 of these brokers, these fields cannot be altered. You must create a new subscription.

Publish/Subscribe Broker: Deregister subscriber using a traditional identity

WebSphere MQ has dropped the concept of a publish/subscribe traditional identity. One consequence is that WebSphere Message Broker and WebSphere Event Broker applications that send a deregister subscriber message can behave differently. They behave differently if the deregister subscriber message for a traditional identity also has named subscriptions. In version 6 of the brokers, the named subscriptions are deleted. In version 7, the same deregister subscriber command does not delete any subscriptions that are named in version 6.

All subscriptions are now named in WebSphere MQ. Any subscriptions that are registered with a traditional identity, and no subscription name, have one generated for them by the queue manager. Only subscriptions with a generated subscription name are deleted when a deregister subscriber command deregisters a traditional identity.

Related reference:

"Publish/Subscribe: Subscription names" on page 656
In IBM WebSphere MQ Version 6.0, a subscription name is unique only within the stream and not across the queue manager. A subscription name must now be unique across the queue manager.

"Publish/Subscribe: Traditional identity" on page 656
The behavior of traditional identities differs between IBM WebSphere MQ Version 6.0 and later versions of IBM WebSphere MQ. In Version 6.0, subscriptions do not require names to identify them; a traditional identity is sufficient. In later versions of every subscription has a name. Subscription names are generated for any Version 6.0 subscriptions without a name that are migrated. Naming subscriptions has no affect on existing Version 6.0 applications. The relationship between traditional identities and subscriptions can result in different behavior.

Publish/Subscribe Broker: Extra topic objects created by migration

Administrative topic objects are created by the migration process. Some of the topic objects might be unnecessary. Although they do not cause any difference in behavior, deleting unwanted topic objects makes publish/subscribe easier to administer.

You are able to construct topic trees in WebSphere Event Broker version 6.0 and WebSphere Message Broker version 6.0 and 6.1. You can do the same in WebSphere MQ. The migration process defines the equivalent topic tree for a queue manager to the tree in the broker being migrated. A difference is that you can also define attributes for particular nodes in the topic tree in WebSphere MQ. You cannot define attributes for particular topic nodes in the brokers. The attributes are stored in administrative topic objects that are associated with particular nodes in the topic tree. Once such attribute is the Access Control List (ACL) for a topic. The initial phase of the migration creates topic objects speculatively. The objects are based on the ACL entries that are defined in the broker. They are created in anticipation that you later run the security commands to create ACLs for the topic objects. When you have resolved what security settings you need, you can delete the topic objects that you do not require.

Related concepts:

"Publish/Subscribe Broker: Access Control List (ACL) migration" on page 371
An automated process migrates the publish/subscribe broker from WebSphere Event Broker and WebSphere Message Broker to WebSphere MQ. The process does not migrate the ACLs automatically. The migration task provides two methods to migrate the ACL to WebSphere MQ.

Related information:

Administrative topic objects
Using an administrative topic object, you can assign specific, non-default attributes to topics.

Topics
A topic is the subject of the information that is published in a publish/subscribe message.

Publish/Subscribe: Interoperation with WebSphere Message Broker

Publish/subscribe does not interoperate between WebSphere Event Broker version 6.0, or WebSphere Message Broker version 6.0, or 6.1, andIBM WebSphere MQ Version 7.0, or later, if you use publish/subscribe clusters, collectives, or broker hierarchies. Publish/subscribe connections cannot be established, and hence subscriptions are not propagated, between the brokers and IBM WebSphere MQ. Applications can exchange messages between IBM WebSphere MQ applications and the brokers by using queues.

You can migrate WebSphere Event Broker version 6.0 and WebSphere Message Broker version 6.0 or 6.1 to IBM WebSphere MQ by using the **migmbbrk** command.

You must migrate all the brokers in a collective to a IBM WebSphere MQ cluster at the same time. Any brokers that remain from WebSphere Message Broker or WebSphere Event Broker will not connect to the migrated brokers.

Publish/Subscribe Broker: Nonpersistent retained publications

WebSphere Message Broker and WebSphere Event Broker version 6.0 and 6.1 publish/subscribe differ from WebSphere MQ publish/subscribe in how they interpret the persistence property for retained publication messages. If you have any nonpersistent retained publications defined, you lose them when you migrate WebSphere Message Broker version 6.0. or 6.1, or WebSphere Event Broker version 6.0 to WebSphere MQ.

In WebSphere Message Broker, the persistence property controls only how the publication is delivered. It controls whether it is persistent or nonpersistent when it is delivered to a subscriber. Unless the subscriber requests persistence, nonpersistent retained publications are delivered as nonpersistent. A persistent or nonpersistent retained publication is stored by the broker until it is explicitly deleted. It is deleted either by an **Delete Publication** message, or because the expiry time for the publication has passed.

In WebSphere MQ, the persistence property controls whether the publication message is delivered to a subscriber as persistent or nonpersistent, as well. It also controls the life span of the retained publication. Unlike in WebSphere Message Broker or WebSphere Event Broker, a nonpersistent retained publication is discarded whenever a nonpersistent message would be deleted. Restarting a queue manager is one event that causes nonpersistent messages to be deleted. Consequently, when a nonpersistent retained publication is migrated, it is discarded at the first queue manager restart.

It is not common for retained publications to be marked as nonpersistent. If any retained publications are marked as nonpersistent, you must recode your application to save these publications before migrating WebSphere Message Broker or WebSphere Event Broker.

Publish/Subscribe Broker: Notification of new or removed subscribers

With WebSphere Event Broker and WebSphere Message Broker version 6.0 and 6.1, an application can monitor subscribers. It can subscribe to topics that publish a message when a subscriber is added to the system and when a subscriber is removed. The brokers in version 7.0 have no direct equivalent. A IBM WebSphere MQ application can request information about subscriptions using MQSC or PCF commands.

In version 6.0 and 6.1, the brokers provide metatopics, which applications can subscribe to, and monitor publish/subscribe. The brokers in version 7.0 can provide no equivalent function using the publish/subscribe broker in IBM WebSphere MQ. IBM WebSphere MQ has no metatopics, nor has it a concept of a subscriber, other than as a connected application that is creating a subscription.

You can call PCF and MQSC commands to query subscriptions and topics. You can also monitor the delivery of publish/subscribe messages.

Related reference:

"Publish/Subscribe: Metatopics" on page 653
Metatopics are a special set of topics recognized by the IBM WebSphere MQ Version 6.0 publish/subscribe broker. Metatopics are no longer supported. Instead you can inquire on the list of topic names, and on individual topics, and subscriptions. If you send a subscription to a metatopic, the subscription is ignored.

Related information:

Monitor Publish/Discarded Publish/Excluded Publish

Publish/Subscribe Broker: Publishing or subscribing to a named stream or subscription point

If a WebSphere Event Broker or WebSphere Message Broker application publishes messages to a named stream or subscription point, you must configure the queue manager with the stream or subscription point.

Related information:

Adding a stream
You can add streams manually so that they coexist with streams migrated from Version 6.0 queue managers.

Adding a subscription point
How to add a subscription point that was not migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker by **`migmbbrk`**. Extend an existing queued publish/subscribe application that you have migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker with a new subscription point.

Streams and topics
Queued publish/subscribe has the concept of a publication stream that does not exist in the integrated publish/subscribe model. In queued publish/subscribe, streams provide a way of separating the flow of information for different topics. In IBM WebSphere MQ Version 6.0, a stream is implemented as a queue, defined at each broker that supports the stream. Each queue has the same name (the name of the stream). From IBM WebSphere MQ Version 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

Subscription points and topics
A subscription point used to request publications from a particular set of publication nodes in WebSphere MQ Event Broker and Message Broker. Named subscription points are emulated by topics and topic objects.

Publish/Subscribe Broker: Multi-topic publications and subscriptions

In WebSphere Message Broker or WebSphere Event Broker, a single command message can publish or subscribe to multiple separate topics. In WebSphere MQ, publish or subscribe can reference only a single topic, or group of topics using a wildcard. To emulate publication or subscription to multiple topics, WebSphere MQ creates multiple publications or subscriptions. As a result after migration, a subscriber might receive multiple publications, where previously the subscriber received only one publication.

The behavior is identical to a WebSphere MQ version 6.0 publish/subscribe application that is migrated to Version 7.5 and publishes or subscribes to multiple topics.

Related reference:

"Publish/Subscribe: Multi-topic publications and subscriptions" on page 653
IBM WebSphere MQ Version 6.0 publish/subscribe has a concept of publishing to multiple topics, or subscribing to multiple topics, in a single publish or subscribe operation. IBM WebSphere MQ publish/subscribe no longer has these concepts. One consequence is that a Version 6.0 application running on a later version queue manager might receive two copies of a publication instead of one.

Publish/Subscribe Broker: Registering multiple subscriptions to the same destination

In WebSphere Event Broker and WebSphere Message Broker version 6.0 and 6.1, you can register subscriptions. A subscription is a registered to a traditional identity. A traditional identity is a combination of queue name, queue manager name, and optional correlation identifier. You can register only one subscription to a traditional identity at a time. Consequently only one subscription can deliver publications to a particular combination of queue name, queue manager name, and optional correlation identifier. In IBM WebSphere MQ no limit is placed on the number of subscriptions that can deliver publications to the same destination.

In WebSphere Event Broker and WebSphere Message Broker version 6.0 and 6.1 the combination of queue name, queue manager name, and optional correlation identifier identifies a traditional identity. IBM WebSphere MQ identifies subscriptions by subscription name, and not by destination. It has no concept of traditional identity. Consequently, in WebSphere Event Broker and WebSphere Message Broker version 7.0, you can create multiple subscriptions to the same destination, that equates to a traditional identity.

Related reference:

"Publish/Subscribe: Subscription names" on page 656
In IBM WebSphere MQ Version 6.0, a subscription name is unique only within the stream and not across the queue manager. A subscription name must now be unique across the queue manager.

"Publish/Subscribe: Traditional identity" on page 656
The behavior of traditional identities differs between IBM WebSphere MQ Version 6.0 and later versions of IBM WebSphere MQ. In Version 6.0, subscriptions do not require names to identify them; a traditional identity is sufficient. In later versions of every subscription has a name. Subscription names are generated for any Version 6.0 subscriptions without a name that are migrated. Naming subscriptions has no affect on existing Version 6.0 applications. The relationship between traditional identities and subscriptions can result in different behavior.

Publish/Subscribe Brokers: Possible loss of data from migrated retained publications with headers in `MQRFH` format

Retained publications in `MQRFH` format might lose data when migrated to IBM WebSphere MQ.

WebSphere Message Broker and WebSphere Event Broker applications that communicate with each other using publish/subscribe command messages, do so in one of two message formats. Broker applications generally use the `MQRFH2` message header, but it is possible that an application might have used the `MQRFH` format. The brokers deliver the message in the format of the subscription and provide automatic conversion to ensure that a subscriber receives messages in the requested format.

The migration of publish/subscribe information from the brokers to IBM WebSphere MQ requests messages in `MQRFH2` format. It is rare for client applications to use messages in `MQRFH` format. However, if an application does create retained messages in `MQRFH` format, it is possible that some truncation of data might occur when migrated. In particular, data passed using the `MQPSStringData` and `MQPSIntData` name/value pairs is not migrated.

The migration process checks for two conditions in the data that is returned from the broker:
- One or more retained messages stored in the broker.
- One or more `MQRFH` subscriptions.

If both these conditions are true, the migration function displays a warning message and writes a warning message in the migration log. The message states that MQRFH retained publications have been migrated with a possible loss of data.

MQRFH2 retained publications are migrated correctly.

Publish/Subscribe Broker: Subscription points

Subscription points are used in WebSphere Message Broker and WebSphere Event Broker to limit the flow of publications to subscribers. Subscription points are migrated to IBM WebSphere MQ by the **migmbbrk** command. After migration, you must add any new subscription points to WebSphere MQ manually.

Subscription points are not part of IBM WebSphere MQ Version 6.0 publish/subscribe. They are now emulated by IBM WebSphere MQ, to migrate publish/subscribe from using the broker in WebSphere Message Broker and Event Broker to IBM WebSphere MQ.

A publication message to a subscription point has a subscription point in the <psc> message folder in IBM WebSphere MQ. In WebSphere Message Broker and WebSphere Event Broker, a subscription point is an attribute of a publication node.

Related information:

Adding a subscription point
How to add a subscription point that was not migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker by **migmbbrk**. Extend an existing queued publish/subscribe application that you have migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker with a new subscription point.

Register Subscriber message

Subscription points and topics
A subscription point used to request publications from a particular set of publication nodes in WebSphere MQ Event Broker and Message Broker. Named subscription points are emulated by topics and topic objects.

Subscription points

Publish/Subscribe Broker: Wildcards

The position of the multilevel wildcard # is restricted to the beginning and end of topic strings in WebSphere Event Broker and WebSphere Message Broker version 6.0 and 6.1. In IBM WebSphere MQ, the multilevel wildcard is valid at any level in the topic string.

Related information:

Version 6.0 wildcard characters

Wildcard schemes
There are two wildcard schemes used to subscribe to multiple topics. The choice of scheme is a subscription option.

IBM WebSphere MQ maintenance tasks

This section describes how to apply and remove maintenance to IBM WebSphere MQ.

Related information:

Getting product fixes

➦ http://www.ibm.com/support/entry/portal/overview/software/websphere/websphere_mq

Applying and removing maintenance level updates on platforms other than z/OS

When you apply and remove maintenance level updates to IBM WebSphere MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix. The following topics describe how to apply and remove fix packs on platforms other than z/OS

About this task

Follow a link in Table 97.

Table 97. Applying and removing maintenance

Apply	**Remove**
AIX	AIX
HP-UX	HP-UX
Linux	Linux
Solaris	Solaris
Windows	Windows

AIX: Applying maintenance level upgrades on IBM WebSphere MQ Version 7.5

How to apply maintenance level updates to IBM WebSphere MQ for AIX using **installp**.

Before you begin

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

About this task

Stop applications using the installation and use the **installp** command, to install maintenance level updates to clients and servers. Alternatively, if the installation is in the default installation location, you can use the *System Management Interface Tool*, SMIT.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.

3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                  ┌─-c─┐
      ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
                  ├─-i─┤
                  └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```

4. Log in as root, or switch to the superuser using the **su** command.
5. Install the update in one of the following ways:
   - Update the whole installation in the default location:

     ```
     installp -agXYd . all
     ```

   - Update selected filesets in the default location:

     ```
     installp -agXYd . list of file sets
     ```

   - Update the whole product in a non-default location using the -R flag:

     ```
     installp -R USIL_Directory -agXYd . all
     ```

   - Update selected filesets in a non-default location using the -R flag:

     ```
     installp -R USIL_Directory -agXYd . list of file sets
     ```

USIL_Directory is the installation parent directory. IBM WebSphere MQ is installed underneath the directory. For example, if /USIL1 is specified, the IBM WebSphere MQ product files are located in /USIL1/usr/mqm. /USIL1/usr/mqm is known as the *MQ_INSTALLATION_PATH*.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Updating installed software on a WebSphere MQ for AIX Version 7.r system

If you want to migrate to a WebSphere MQ for AIX, Version 7.r system (where r in this situation is 5) from an earlier version of the WebSphere MQ system using SMIT, you must perform certain tasks.

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1.6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can use the *System Management Interface Tool*, SMIT, only if the installation is in the default installation location.

You must first end all WebSphere MQ activity on the target machine.

This migration procedure applies only to migration from an earlier version of a WebSphere MQ system. If you are migrating from an earlier version of WebSphere MQ or MQSeries for AIX, you are advised to uninstall your current version before installing the new version.

Migration from an earlier version of WebSphere MQ for AIX involves updating any currently installed file sets. It also involves installing any new file sets that might be required.

To update currently installed file sets:

1. Go into SMIT for root authority. From the shell, enter:
   ```
   smit
   ```
2. Select the device appropriate for your installation using the following sequence of windows:
   ```
   Software Installation and Maintenance
     Install and Update Software
       Update Installed Software to Latest Level (Update All)
   ```
   Alternatively, you can use the **fastpath** command to select the appropriate device:
   ```
   smitty update_latest
   ```
3. Click **List** to display the Single Select List window.
4. Click **/dev/cd0 (CD Drive)**.
5. Click **OK** to display the parameters for **Update All**.
6. Update all previously installed software for WebSphere MQ by selecting the **_update_all** option in the **Software to update** field.
7. Click **Enter**.
8. Click **OK** in the confirmation window to start updating the software.

When all previously installed file sets have been updated to the latest level, you can install any additional file sets.

AIX: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5

How to restore a previous maintenance level using the *System Management Interface Tool* (SMIT).

Before you begin

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

About this task

You can back out maintenance updates and restore your system to the previous maintenance/install level, for any component of IBM WebSphere MQ for AIX that is in the **APPLIED** state.

Use the following command to display the current state of the IBM WebSphere MQ for AIX filesets:

```
lslpp [ -R usil ] -l "mqm*"
```

To back out a maintenance update, as the user root, issue the command:

```
installp [ -R usil ] -r "mqm*"
```

Otherwise:

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
             ┌─-c─┐
►►──endmqm───┼─-w─┼──QmgrName──────────────────────────────────────────►◄
             ├─-i─┤
             └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Open the appropriate **smit** panel using this sequence:

```
Software Installation and Maintenance
  Software Maintenance and Utilities
    Reject Applied Software Updates (Use Previous Version)
```

   Alternatively, use a fast path command, `smit[ty] install_update`.
6. Complete the **SOFTWARE** name field.

   Enter `mqm*` to restore all applicable file set updates to your installation.

   Note: If an option to restore only selected file set updates for IBM WebSphere MQ for AIX appears, avoid it. The option results in all applicable file set updates for the maintenance update being restored.
7. Click **Enter** to reject the current maintenance level and reinstate the previous maintenance or install level.

   a. Accept displayed default values for all other fields

   b. Dismiss the confirmation message

   The reject process starts. While the command runs, it displays progress messages terminating with an **Install Summary** table.

   a. Check the table to see which components of IBM WebSphere MQ for AIX have been rejected

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

HP-UX: Applying maintenance level updates on IBM WebSphere MQ Version 7.5

How to apply maintenance level updates to IBM WebSphere MQ for HP-UX using **swinstall** .

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.
3. You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

About this task

1. If you want to install both the base package and the maintenance update packages, install the base package separately first. Then install the maintenance update packages.
2. If you are using the interactive installer, click **Options > Change Options**. Then clear the **autoselect dependencies when marking software** check box before selecting the maintenance update package for installation.
3. Error messages might be seen when running **swinstall** , even when successfully updating an installation.

   There are two approaches you can take to handling errors in the application of maintenance.

   a. Aim for an error-free update by applying maintenance only to those components that are installed.
   b. Apply the whole maintenance package and check the error logs, error by error, ignoring the insignificant errors.

   Both approaches are described.

   Many of the insignificant errors are caused by **swinstall** trying to apply updates to components that are not installed. Consider whether there are any significant errors reported with the insignificant ones.

   - The following errors might not indicate a serious problem. They are written to the console, or to the **swinstall** panel.

     ```
     ERROR:   "hpux11.mycompany.com:/":
     The software dependencies for 15 products or filesets cannot be resolved.
     ```

     ```
     ERROR:   "hpux11.mycompany.com:/":
     17 filesets were determined to be skipped in the analysis phase.
     The execution phase failed for "hpux11.mycompany.com:/".
     Analysis and Execution had errors.
     ```

   - The following errors might not indicate a serious problem. They are written to the swjob output for a **swinstall** session.

     ```
     ERROR:   17 of 20 filesets had Errors.
     3 of 20 filesets had no Errors or Warnings.
     ```

```
ERROR:   The Execution Phase had errors.
See the above output for details.
```

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                 ┌─-c─┐
      ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
                 ├─-i─┤
                 └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.
   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Log in as root, or switch to the superuser using the **su** command.
5. Make your current directory the location of the *Service_update_package*.

   The file name of the *Service_update_package* follows the pattern `hp-Uxxxx.v11`. You must prefix *Service_update_package* with the absolute path to the installation file. To save typing, construct the path using the `$PWD` variable.

6. Run the HP-UX command **swlist** l=*MQ_INSTALLATION_PATH* MQSERIES to list all of the IBM WebSphere MQ components that are installed.
7. Decide whether to install the updates interactively, and if you want to control which components are updated.

   You can update in the following ways:

   - Silently update all the installed IBM WebSphere MQ components by installing the whole maintenance package.

     ```
     swinstall -s $PWD/service_update_package
      MQSERIES,l=MQ_INSTALLATION_PATH
     ```

     The **swinstall** command attempts to find an installed component for every component in the update package, and updates it. **swinstall** writes out error messages for components that it cannot find.
   - Silently update some IBM WebSphere MQ components by installing only the required updates from the maintenance package.

     If you specify *update_components* correctly, the update procedure can be error-free. **swinstall** only updates components that you have listed and components that are dependent on components you have listed.

     a. Using the list of installed IBM WebSphere MQ components, create a space separated list of the components you want to update (*update_components*). This list requires the installation path of each component to be specified, in the form: *component*,l=*MQ_INSTALLATION_PATH*
     b.
        ```
        swinstall -s $PWD/service_update_package
         update_components
        ```
   - Interactively update some IBM WebSphere MQ components from the maintenance package, selecting only the update components that are required.

     ```
     swinstall -s $PWD/service_update_package
     ```

     a. Open **MQSERIES** and mark the update components you want to apply. Correctly marked, there are no errors when the updates are applied. The installer resolves dependencies automatically.
     b. Select **Actions > Change Product Location** to change the IBM WebSphere MQ installation you intend to update.
     c. Select **Actions > Install**. The log file tells you if there are any problems that need fixing.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

HP-UX: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5

How to restore a previous maintenance level by using **swremove**.

Before you begin

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

Important: pax and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

      ```
                     ┌─-c─┐
      ►►──endmqm─────┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
                     ├─-i─┤
                     └─-p─┘
      ```

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.
   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Log in as root, or switch to the superuser using the **su** command.
5. Run the **swremove** command to remove the maintenance package from the system.

   For example, to remove the 7.R.0.1 maintenance level, use the command:

   ```
   swremove MQSERIES,r=7.R.0.1,l=MQ_INSTALLATION_PATH
   ```

   where:

   - `R` is the number of the Release
   - *`MQ_INSTALLATION_PATH`* is the installation path for IBM WebSphere MQ

Details of the **swremove** command can be found in the *HP-UX Administration Guide* or by using the **man swremove** command.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Linux: Applying maintenance level updates on IBM WebSphere MQ Version 7.5

How to apply maintenance level updates to IBM WebSphere MQ for Linux using RPM.

Before you begin

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

About this task

Maintenance level updates are delivered in the form of Red Hat Package Manager (RPM) update images, which are applied using the RPM installation tool.

Additional disk space is required for the update images to allow maintenance level updates to be removed and the previous level restored. The updated files are kept in `MQ_INSTALLATION_PATH/maintenance` directory. Do not delete or move this directory or the files it contains.

`MQ_INSTALLATION_PATH` represents the high-level directory in which IBM WebSphere MQ is installed.

Updates are cumulative. You can apply your chosen update directly, without applying any previous updates first. The maintenance level updates might contain updates for one or more packages. You must apply those parts of an update that correspond to the packages that are applied in your installation.

Important: Although it is possible to install a fix pack at the same level as an installation performed from a manufacturing refresh image at that level, you should not attempt this process. Installing a fix pack at the same level as the one already on your system, can leave the package management database of your system in an inconsistent state with respect to the installation of IBM WebSphere MQ.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
               ┌─-c─┐
►►──endmqm─────┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
               ├─-i─┤
               └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Change into the directory containing the maintenance packages.
6. Run the `ls` command to list the available updates.

For example, if there are level 1 maintenance updates for the Runtime, SDK and Server packages, you see the following:

```
MQSeriesRuntime-Uxxxx-7.R.0-1.i386.rpm
MQSeriesSDK-Uxxxx-7.R.0-1.i386.rpm
MQSeriesServer-Uxxxx-7.R.0-1.i386.rpm
```

where `R` is the number of the Release.

7. Run the **rpm** command to find out which packages are installed on your server.

Enter the following command:

```
rpm -qa | grep MQSeries
```

For example, if you have a minimum IBM WebSphere MQ installation and SDK component, at level 0, the **rpm** command returns:

```
MQSeriesRuntime-7.R.0-0
MQSeriesSDK-7.R.0-0
MQSeriesServer-7.R.0-0
```

where R is the number of the Release.

8. If this fix pack is to be upgraded on an installation, other than the first installation on the system, run the **crtmqfp** command to create and use a unique set of packages to install on the system. Note, that if this is the first, or only, IBM WebSphere MQ installation on the system, you can ignore this step. You must install the **pax** command in order for the **crtmqfp** command to run on Linux.
   a. Run the command `./crtmqfp <suffixname>` where *suffixname* is the same as the suffix used during renaming of the base level IBM WebSphere MQ installation.
   b. Set your current directory to the location specified when the **crtmqfp** command completes. This directory is a subdirectory of `/var/tmp/mq_rpms`, in which the unique set of packages is created. The packages have the suffix value contained within the filename.

      For example, if you used suffix 1 during repackaging of the base level IBM WebSphere MQ installation, enter the command: **crtmqfp** 1.

      There is now a subdirectory named `/var/tmp/mq_rpms/1/xxxx`, and the packages will be renamed, for example, from `MQSeriesRuntime-7.R.0-1.xxxx.rpm` to `MQSeriesRuntime_1-7.R.0-1.xxxx.rpm`. Where R is the number of the Release.
9. Run the **rpm** command to apply all available updates for the packages you have on your system:
   - To update an installation in the default location, `/opt/mqm`:

     ```
     rpm -ivh MQSeriesRuntime-Uxxxx-7.R.0-1.i386.rpm
       MQSeriesSDK-Uxxxx-7.R.0-1.i386.rpm
       MQSeriesServer-Uxxxx-7.R.0-1.i386.rpm
     ```

     where R is the number of the Release.
   - To update an installation in a custom location, specify the **rpm** `prefix` option:

     ```
     rpm --prefix /opt/customLocation -ivh MQSeriesRuntime-Uxxxx-7.R.0-1.i386.rpm
       MQSeriesSDK-Uxxxx-7.R.0-1.i386.rpm
       MQSeriesServer-Uxxxx-7.R.0-1.i386.rpm
     ```

     where R is the number of the Release.

   You must apply all packages in a maintenance update that correspond to those packages that are currently installed on your system.
10. Repeat step 7 on page 686 to list the packages that are now available.

    The Runtime, SDK, and Server packages are now at level 1:

    ```
    MQSeriesRuntime-7.R.0-0
    MQSeriesSDK-7.R.0-0
    MQSeriesServer-7.R.0-0
    MQSeriesRuntime-Uxxxx-7.R.0-1
    MQSeriesSDK-Uxxxx-7.R.0-1
    MQSeriesServer-Uxxxx-7.R.0-1
    ```

    where R is the number of the Release.

    Note:

    After the installation of IBM WebSphere MQ fix packs, if you run the `rpm-verify` or `rpm -V` command, it does not return the correct results. It produces spurious results relating to missing files in *`MQ_INSTALLATION_PATH`*/maintenance.

    This error message can be ignored because it is a known limitation in the IBM WebSphere MQ fix pack installation code. For further information about this error, see WebSphere MQ Fix Pack install errors - Linux reports errors

What to do next

For further information about using RPM to install software packages, see your Linux documentation.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Linux: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5

How to remove updates and restore the previous maintenance level using **RPM**.

Before you begin

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

About this task

When maintenance is applied, the original versions of replaced files are saved to allow the updates to be removed if necessary. To restore the previous maintenance level, run an Red Hat Package Manager, RPM, uninstall command for all the packages that were updated by the maintenance package as follows:

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
          ┌─-c─┐
►►─endmqm─┼─-w─┼─QmgrName──────────────────────────────────────────────►◄
          ├─-i─┤
          └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Run the **rpm** command to find out which packages are installed on your server.

   Enter the following command:

```
rpm -qa | grep MQSeries
```

   Using the example given in "Linux: Applying maintenance level updates on IBM WebSphere MQ Version 7.5" on page 685, returns:

```
MQSeriesRuntime-7.R.0-0
MQSeriesSDK-7.R.0-0
MQSeriesServer-7.R.0-0
MQSeriesRuntime-Uxxxx-7.R.0-1
MQSeriesSDK-Uxxxx-7.R.0-1
MQSeriesServer-Uxxxx-7.R.0-1
```

   where R is the number of the Release.
6. Run the **rpm** command to remove all the updates applied at level 1.

   Enter the following commands:

```
rpm -ev MQSeriesRuntime-Uxxxx-7.R.0-1 MQSeriesSDK-Uxxxx-7.R.0-1
  MQSeriesServer-Uxxxx-7.R.0-1
```

   where R is the number of the Release.
7. Repeat step 5 to check that the ptf packages have been removed, leaving only the original install packages:

```
MQSeriesRuntime-7.R.0-0
MQSeriesSDK-7.R.0-0
MQSeriesServer-7.R.0-0
```

   where R is the number of the Release.

What to do next

For further information on using RPM to install software packages, see your Linux documentation.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Solaris: Applying maintenance level updates on IBM WebSphere MQ Version 7.5

How to apply maintenance level updates to IBM WebSphere MQ for Solaris using **pkgadd**.

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.
3. You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

About this task

Stop applications using the installation and use **pkgadd** to install maintenance.

Important: Although it is possible to install a fix pack at the same level as an installation performed from a manufacturing refresh image at that level, you should not attempt this process. Installing a fix pack at the same level as the one already on your system, can leave the package management database of your system in an inconsistent state with respect to the installation of IBM WebSphere MQ.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
            ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────────►◄
            ├─-i─┤
            └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Change into the directory containing the maintenance packages.
6. Run the **crtmqfp** command to create and use a unique set of packages to install on the system, if this fix pack is to be upgraded on a installation that is not the first installation on the system. This command creates and uses a unique set of packages to install on the system.
   a. Run the command **crtmqfp** mqm-<*suffixname*> where *suffixname* is the same as the suffix used during renaming of the base level IBM WebSphere MQ installation. Note that this command creates a full copy of the installation packages in a subdirectory of `/var/tmp`.
   b. Set your current directory to the location specified when the **crtmqfp** command completes. This directory is a subdirectory of `/var/spool`, in which the unique set of packages is created. The packages have the suffix value contained within the filename.
7. Enter the following command:

   ```
   pkgadd -d packagename
   ```

   where `packagename` corresponds to the image file name. For example:

   ```
   mqm-U1234.img
   ```

   For further information about using **pkgadd** to install software packages, see the Solaris documentation.
8. Follow the on-screen instructions.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Solaris: Applying maintenance level updates on IBM WebSphere MQ in non-interactive mode

You can install IBM WebSphere MQ for Solaris non-interactively by creating a response file and an admin file.

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.
3. You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

About this task

Stop applications using the installation and use **pkgadd** to install maintenance.

Important: Although it is possible to install a fix pack at the same level as an installation performed from a manufacturing refresh image at that level, you should not attempt this process. Installing a fix pack at the same level as the one already on your system, can leave the package management database of your system in an inconsistent state with respect to the installation of IBM WebSphere MQ.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
          ┌─-c─┐
►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────────►◄
          ├─-i─┤
          └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Change into the directory containing the maintenance packages.
6. Create the non-interactive install response file using the **pkgask** command:

```
pkgask -d <location_to_image>/packagename -r response.txt mqm-07-05-00-02
```

where packagename corresponds to the image file name, for example mqm-U200403.img, response.txt is the name of the response file to create, and mqm-07-05-00-02 is the fix pack package name.

7. Find the admin_file from the server installation media located at <install_media>/silent/admin or create an admin_file in the following format:

```
mail=
instance=unique
partial=ask
runlevel=ask
idepend=ask
rdepend=ask
space=ask
setuid=nocheck
conflict=nocheck
action=nocheck
basedir=default
```

8. Run the **pkgadd** command to apply the maintenance level update IBM WebSphere MQ for Solaris in non-interactive mode:

```
pkgadd -v -n -r response.txt -a admin_file -d <location_to_image>/packagename mqm-07-05-00-02
```

where admin_file is a path qualified name of the admin file you created, and mqm-07-05-00-02 corresponds to the fix pack package being installed.

9. Follow the on-screen instructions.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Solaris: Restoring the previous maintenance level on IBM WebSphere MQ Version 7.5

How to restore a previous maintenance level by stopping WebSphere MQ and using **pkgrm**.

Before you begin

If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.

Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

About this task

When maintenance is applied, the original versions of replaced files are saved to allow the updates to be removed if necessary. To restore the previous maintenance level, run **pkgrm** command for all the packages that were updated by the maintenance package as follows:

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
          ┌─-c─┐
►►─endmqm─┼─-w─┼─QmgrName─────────────────────────────────────────►◄
          ├─-i─┤
          └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Run the **pkgrm** command to remove the latest maintenance update from the system:

   ```
   pkgrm packagename
   ```

   packagename is the name of the package that you want to remove; for example, `mqm-07-R-00-01.img`, where `R` is the number of the Release.

   Details of the **pkgrm** command can be found in the Solaris documentation, or by using the **man pkgrm** command.

   If you do not know the name of the package to remove, try listing the packages that are installed using the following command: `pkginfo | grep mqm`

Note: Ignore any error messages of the form `<shared pathname not removed>`.

What to do next

If you have installed a IBM WebSphere MQ MQI client, and the client was updated after installing the maintenance level that is being removed, you must specifically update your IBM WebSphere MQ MQI client installation again, after the maintenance level has been removed

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Windows: Applying maintenance level upgrades on IBM WebSphere MQ Version 7.5

How to apply maintenance level updates to WebSphere MQ for Windows.

Before you begin

1. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

2. You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do steps 1 on page 685, 2 on page 685, 5 on page 686, and 6 on page 686 in the following maintenance procedure.
3. Download the maintenance package from the IBM WebSphere MQ Support website.
4. If User Account Control (UAC) is enabled, the user who does the installation must have Administrative authority. You must elevate any command or command prompt by selecting **Run as Administrator**. If you do not, the error `AMQ4353` is written in the installation log

Procedure

1. Log on as an Administrator.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b. Run the **`MQSC`** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo DISPLAY LSSTATUS(*) STATUS | runmqsc QmgrName
      ```

   c. Run the **`endmqm`** command to stop each running queue manager associated with this installation.

      ```
                  ┌─-c─┐
      ►►──endmqm──┼─-w─┼──QmgrName──────────────────────────────────────────►◄
                  ├─-i─┤
                  └─-p─┘
      ```

      The **`endmqm`** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **`endmqm`** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

      Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

   d. Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```

4. Stop the IBM WebSphere MQ service for the installation.
   a. Right-click the **WebSphere MQ** icon in the taskbar > click **Stop WebSphere MQ**.
5. Load and apply the maintenance files for server or client installations:
   - Interactively:
     a. Open the folder where the maintenance package has been extracted.
     b. Right-click on the maintenance program and select **Run as administrator** to start the loading process.
     c. Choose your installation language, and click **OK** .
     d. Continue to follow the instructions on screen.

        If you choose to load the files without applying them to an installation, you can apply the files later, as described in step 6 on page 698
   - Silently:
     a. Open the folder where the maintenance package has been extracted.
     b. Modify the response file, `silent_install.resp` . For details on the properties you can specify in the response file, see Table 98

Table 98. Properties used to install or uninstall a maintenance update

Property	Value	Description
MQPLOG	*path\file_name*	Pass a valid path to specify the log to be used during installation/uninstallation, for example MQPLOG="C:\TEMP\UPDATEINSTALL.LOG"
MQPINSTALLATIONNAME	*Installation name*	The name of the installation to maintain. If there is only 1 installation (of any level) on the machine, this argument can be safely omitted. If there is more than 1 installation on the machine, amqicsdn.exe checks the value of MQPINSTALLATIONNAME. If one is not supplied, or the one that is supplied is unsuitable, then a GUI selection box appears. This selection box provides a list of installations to which this fix pack is applicable. If none are applicable, then amqicsdn.exe issues error message AMQ4781 and ends.
MQPBACKUPPATH	*path*	Specifies the directory to back up into during installation, for example MQPBACKUPPATH="C:\BACKUP" The directory, and any intermediate directories, you specify must already exist. If any one of the directories does not already exist, the install fails.
MQPREBOOT	0|1	Specifies what to do when a reboot is required, for example MQPREBOOT=1. If no value is supplied, you are prompted for what to do next. If MQPREBOOT is set to 0, a reboot is suppressed If MQPREBOOT is set to 1, the reboots go ahead without prompting.
MQPINUSEOK	0|1	Specifies whether to continue even if a file is found to be currently locked by another application. If you choose to continue even if a file is locked by another application, then you must reboot to complete fix pack installation. If no value is supplied, or if MQPINUSEOK is set to 0, the installation fails if files are found to be in use by other applications. If MQPINUSEOK is set to 1, the installation is deferred until you reboot.

c. Open an elevated command prompt in the directory where the maintenance program was extracted.

d. Start the silent loading by entering the following command:

```
executableName -f responseFile
```

where:

- *executableName* is the name of the maintenance package. For example, for version 7.1, fix pack 1: `7.1.0-WS-MQ-Windows-FP0001.exe` .
- *responseFile* is the full path and name of the response file.

6. Optional: Apply the maintenance to other server or client installations on the system:
   - Interactively:

     a. From the Windows start menu, select **Start** > **Programs** > **IBM WebSphere MQ** > **Apply Fix Pack 7.1.n.n**.

     b. Continue to follow the instructions on screen.
   - Silently:

     a. Open an elevated command prompt and navigate to the directory where the maintenance program was loaded. By default, the path is `C:\Program Files (x86)\IBM\source\WebSphere MQ` *7.1.n.n*

     b. Enter the following command:

     ```
     amqicsdn MQPINSTALLATIONNAME=name MQPSILENT=1
     ```

     where *name* is the name of the installation that you want to apply maintenance to.

     You can add other properties to the command, as listed in Table 98 on page 697.

Applying maintenance to a client installation using MSI

7. As an alternative method for applying silent maintenance to client installations, you can use the MSI command **`msiexec`** from the command line.
   - To upgrade a client system with only a single installation, you can use a command similar to the following example, where PATH is the path of the client installation media:

   ```
   msiexec /i "PATH\Windows\MSI\IBM WebSphere MQ.msi" /l*v <install_log_path>
    /q TRANSFORMS="1033.mst" REINSTALL=ALL REINSTALLMODE=vomus
   ```
   - For a multi installation client system, you can upgrade by using a command similar to the following example:

   ```
   msiexec /i "PATH\Windows\MSI\IBM WebSphere MQ.msi" /l*v <install_log_path>
    /q TRANSFORMS=":InstanceId2.mst;1033.mst" REINSTALL=ALL REINSTALLMODE=vomus
   ```

What to do next

On a server installation, you must restart the IBM WebSphere MQ taskbar application manually after the maintenance application completes.

The IBM WebSphere MQ service is restarted automatically on the server, but the taskbar application is not restarted for any logged in sessions. Start the taskbar application in one of three ways:

1. Start the taskbar application manually from the start menu.
2. Log off and log back on again.
3. Run the command:

```
MQ_INSTALLATION_PATH\bin\amqmtbrn.exe -Startup
```

Related reference:

"Windows: WebSphere MQ Installation affected by User Account Control on Windows systems" on page 668
On some Windows systems, installation of WebSphere MQ is interrupted at various points by User Account Control (UAC). You must acknowledge the Windows UAC prompt to allow installation to run with elevated authority. You must run silent installation from an elevated command prompt.

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Windows: Restoring the previous backup version on IBM WebSphere MQ Version 7.5

How to remove updates and restore the previous maintenance level using the Windows installer

Before you begin

1. If you are running on a server with multiple IBM WebSphere MQ installations, that is, IBM WebSphere MQ Version 7.0.1, Fix Pack 6 (or later), you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.
2. You can apply and remove maintenance from a IBM WebSphere MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 on page 685 to 3 in the following maintenance procedure.
3. If User Account Control (UAC) is enabled, the user who does the installation must have Administrative authority. You must elevate any command or command prompt by selecting **Run as Administrator**. If you do not, the error `AMQ4353` is written in the installation log

About this task

If you applied maintenance to IBM WebSphere MQ, you can restore IBM WebSphere MQ to a previous level of maintenance. If you installed IBM WebSphere MQ at a particular maintenance level, a *Manufacturing Refresh*, you cannot restore IBM WebSphere MQ to an earlier maintenance level.

Procedure

1. Log on as an Administrator.
2. Stop all applications using the IBM WebSphere MQ installation.

   If you use the IBM WebSphere MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM WebSphere MQ installation.
   a. Run the **`dspmq`** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM WebSphere MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.
   b. Run the **`MQSC`** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo DISPLAY LSSTATUS(*) STATUS | runmqsc QmgrName
```

c. Run the **endmqm** command to stop each running queue manager associated with this installation.

```
             ┌─-c─┐
►►──endmqm───┼─-w─┼──QmgrName─────────────────────────────────────────────────────►◄
             ├─-i─┤
             └─-p─┘
```

The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM WebSphere MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM WebSphere MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM WebSphere MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM WebSphere MQ shared libraries from the installation loaded prevent you applying IBM WebSphere MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep a IBM WebSphere MQ shared library loaded.

d. Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Stop the IBM WebSphere MQ service for the installation.

   a. Right-click the **WebSphere MQ** icon in the taskbar > click **Stop WebSphere MQ**.

5. Remove the maintenance interactively, or silently using a command.

   - Interactively:

     a. For each installation of IBM WebSphere MQ that has had maintenance applied, you are presented with one of the following icons in the Windows start menu:

        1) **Start > Programs > IBM WebSphere MQ > Remove Refresh Pack 7.5.n.0** ***(installation name)***

        2) **Start > Programs > IBM WebSphere MQ > Remove Fix Pack 7.5.n.n** ***(installation name)***

     b. Select the installation you want to maintain and click **Remove** to start the process.

     This returns the installation to the state it was in before the maintenance package was applied.

   - Silently:

     a. Open an elevated command prompt and enter the following command:

        ```
        amqicsdn.exe MQPINSTALLATIONNAME=name MQPUNINST=1 MQPSILENT=1
        ```

        where *name* is the name of the installation that you want to remove maintenance from.

        You can add other properties to the command, as listed in Table 99 on page 701.

Table 99. Properties used to install or uninstall a maintenance update

Property	Value	Description
MQPLOG	*path\file_name*	Pass a valid path to specify the log to be used during installation/uninstallation, for example MQPLOG="C:\TEMP\UPDATEINSTALL.LOG"
MQPINSTALLATIONNAME	*Installation name*	The name of the installation to maintain. If there is only 1 installation (of any level) on the machine, this argument can be safely omitted. If there is more than 1 installation on the machine, amqicsdn.exe checks the value of MQPINSTALLATIONNAME. If one is not supplied, or the one that is supplied is unsuitable, then a GUI selection box appears. This selection box provides a list of installations to which this fix pack is applicable. If none are applicable, then amqicsdn.exe issues error message AMQ4781 and ends.
MQPBACKUPPATH	*path*	Specifies the directory to back up into during installation, for example MQPBACKUPPATH="C:\BACKUP" The directory, and any intermediate directories, you specify must already exist. If any one of the directories does not already exist, the install fails.
MQPREBOOT	0|1	Specifies what to do when a reboot is required, for example MQPREBOOT=1. If no value is supplied, you are prompted for what to do next. If MQPREBOOT is set to 0, a reboot is suppressed If MQPREBOOT is set to 1, the reboots go ahead without prompting.
MQPINUSEOK	0|1	Specifies whether to continue even if a file is found to be currently locked by another application. If you choose to continue even if a file is locked by another application, then you must reboot to complete fix pack installation. If no value is supplied, or if MQPINUSEOK is set to 0, the installation fails if files are found to be in use by other applications. If MQPINUSEOK is set to 1, the installation is deferred until you reboot.

6. Optional: If you no longer need the maintenance files that were loaded onto the system before maintenance was applied, you can remove them using **Add/Remove programs** or **Programs and Features** from the Control Panel. If you want to remove a maintenance file silently, run the following command:

   ```
   <patch_install_files>\_IBM WebSphere MQ (fix pack 7.5.0.1 files)_installation\Change IBM WebSphere MQ (fix pack 7.5.0
   ```

   where <patch_install_files> is the installation directory where maintenance files are installed.

   By default, this directory is c:\Program Files [(x86)]\IBM\source\WebSphere MQ 7.5.0.1

   Notes:

   a. Run the command from outside the directory, otherwise the directory is not removed.
   b. If you omit **-i silent**, the command initiates the Graphical User Interface installer.

What to do next

On a server installation, you must restart the IBM WebSphere MQ taskbar application manually after the maintenance application completes.

The IBM WebSphere MQ service is restarted automatically on the server, but the taskbar application is not restarted for any logged in sessions. Start the taskbar application in one of three ways:

1. Start the taskbar application manually from the start menu.
2. Log off and log back on again.
3. Run the command:

   ```
   MQ_INSTALLATION_PATH\bin\amqmtbrn.exe -Startup
   ```

Related information:

dspmq

Stopping a queue manager
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Applying maintenance level upgrades to multi-instance queue managers

Use multi-instance queue managers to reduce the outage caused by applying maintenance updates. Follow these steps to apply maintenance to a multi-instance queue manager.

Before you begin

Maintenance is applied to the WebSphere MQ installation on a server and not to individual queue managers. You must stop all the queue managers, and any WebSphere MQ service, on a server before you apply maintenance.

If you want a queue manager to keep running while maintenance is applied, you must configure it as a *multi-instance* queue manager, and have a standby instance running on another server. If a queue manager is an existing single instance queue manager, you must convert it to a multi-instance queue manager. See the topic, Multi-instance queue managers for prerequisites and guidance how to create a multi-instance queue manager.

You can create a multi-instance queue manager from v7.0.1 onwards. If you are running multi-instance queue managers, you then can apply a maintenance update to a *running* queue manager by switching the active instance to a different server.

Typically active and standby installations are maintained at the same maintenance level. Consult the maintenance instructions for each upgrade. Consult the instructions to see if it is possible to run the active and standby instances at different maintenance levels. Check whether fail over from higher to lower, or only lower to higher maintenance level is possible.

The instructions for applying a maintenance update might require you to stop a multi-instance queue manager completely.

If you have a primary server for running active queue manager instances, and a secondary server that runs standby instances, you have a choice of updating the primary or secondary server first. If you update the secondary server first, you must switch back to the primary server when both servers have been updated.

If you have active and standby instances on several servers, you must plan in what order you update the servers to minimize the disruption caused by ending the active instances on each server you update.

About this task

Combine the steps in this task with the maintenance update procedure for applying maintenance to a WebSphere MQ server installation.

Procedure

1. Where the maintenance update procedure instructs you to stop all running queue managers, or quiesce WebSphere MQ do the following instead:

   The maintenance update procedure varies by platform; see "Applying and removing maintenance level updates on platforms other than z/OS" on page 676.

   a. If the queue manager is running as standby:
      - On Windows, UNIX, and Linux platforms, end the standby with the **endmqm** -x *QMgrName* command.

   b. If the queue manager is running as the active instance:
      - On Windows, UNIX, and Linux platforms, end the instance and transfer control to the standby instance with the **endmqm** command. For example, **endmqm** *-shutdown_option* -s *QMgrName*, where *-shutdown_option* is an optional parameter specifying the type of shutdown. For more information, see endmqm.

        If there is no standby instance running, the command fails, and you must start a standby instance on a different server.

   c. If a queue manager is running as a single instance queue manager, you have no alternative but to stop the queue manager before applying the maintenance update.

   When you complete this step, no queue manager instances are left running on the server you intend to update.
2. Continue with the maintenance update procedure, following the step to issue the **endmqm** command, or quiesce WebSphere MQ and apply maintenance to the WebSphere MQ server.
3. When you have completed the maintenance update, restart all the queue managers on the WebSphere MQ server, permitting standby instances:
   - On Windows, UNIX, and Linux platforms, use the **strmqm** -x *QmgrName* command.
4. Repeat the procedure on the standby server, to update its maintenance level.
5. If necessary, switch the active instances back to the primary servers:
   - On Windows, UNIX, and Linux platforms, use the **endmqm** *-shutdown_option* -s *QMgrName* command, and the restart the instances using the **strmqm** -x *QmgrName* command.

Related tasks:

"Applying and removing maintenance level updates on platforms other than z/OS" on page 676
When you apply and remove maintenance level updates to IBM WebSphere MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix. The following topics describe how to apply and remove fix packs on platforms other than z/OS

Migrating queue managers to new-function fix packs

This scenario illustrates running different levels of queue manager from a single installation using new-function fix packs. New function fix-packs are available on platforms other than z/OS. It contrasts migrating a queue manager to new command levels in new-function fix packs, to migrating a queue manager to a new command level in a new release. The scenario explains the relationship between new-function fix packs and maintenance fix packs.

Before you begin

In this section, IBM WebSphere MQ Version 7.1 is used as the current release, and the release is denoted by r; the subsequent release is denoted by R.

The scenario starts with a single installation of IBM WebSphere MQ Version 7.1, `Inst_1`. `Inst_1` is the primary installation; see Figure 102 on page 708. For illustration, there are two queue managers, `QM1` and `QM2` . `QM1` stays at the `7r0` command level, `QM2` moves to the highest command level available.

The use of version numbers and command levels is illustrative, and does not imply anything about future releases.

About this task

Figure 101 on page 705 has time advancing down the Y-Axis, as new fix packs are released. On the X-Axis are different command levels. As a queue manager is migrated to a new command level, it shifts across the diagram. Each column represents the fix levels a queue manager at a particular command level can run at.

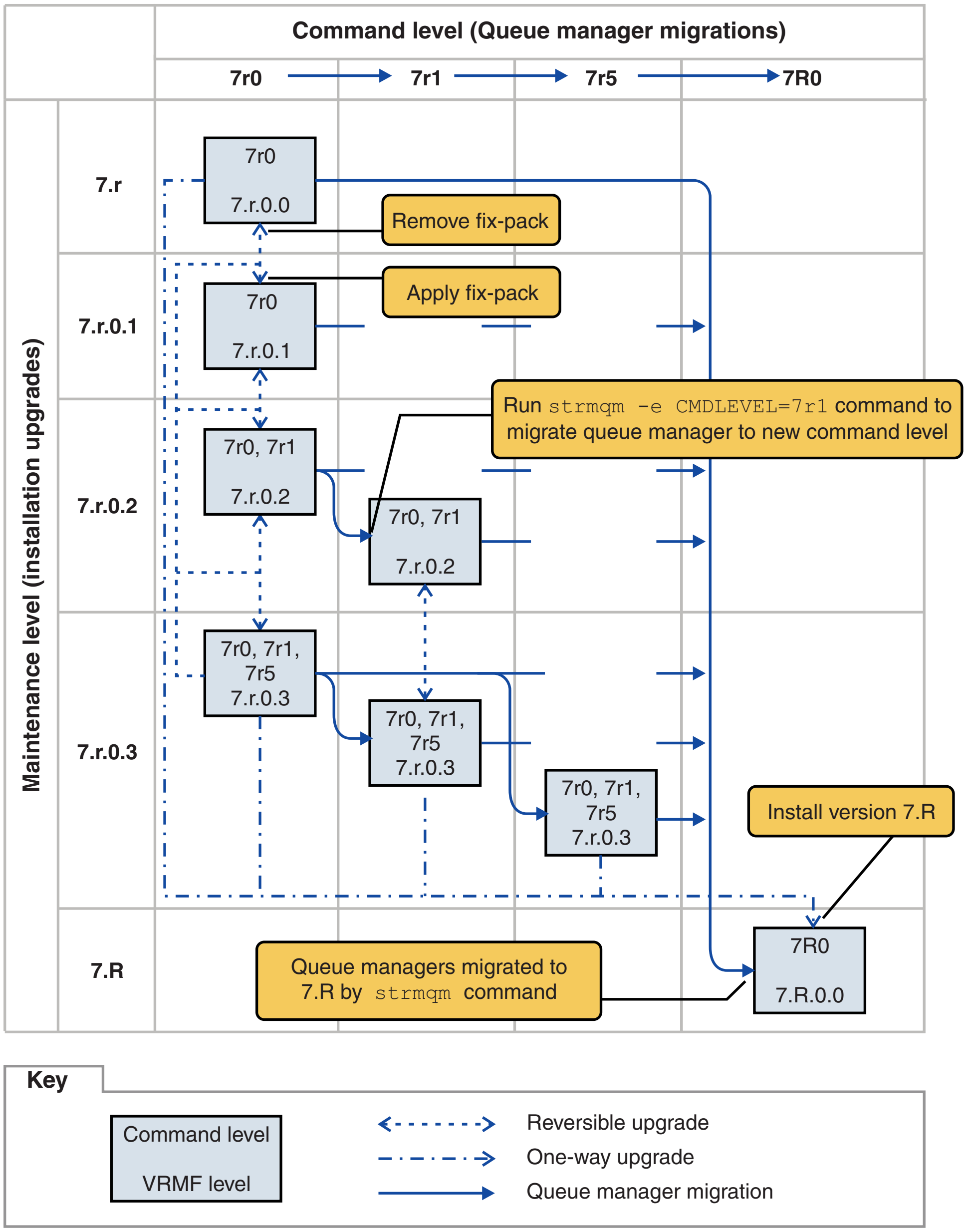


Figure 101. Migration of queue managers to new command levels using new-function fix packs

Figure 101 is a little complicated, but it captures lots of details about new-function fix packs to help you remember them. The steps in the task explain the details in the figure. Some of the principle features of Figure 101 are explained in the following list:

Maintenance level and Command level

The maintenance level is a fix pack with a V.R.M.F. code; see “The version naming scheme for WebSphere MQ on platforms other than z/OS” on page 324. V.R.M.F codes are one to four digits, always punctuated by periods. Trailing zeros are sometimes omitted in descriptions, but never when a V.R.M.F code is used to label a fix pack. Version 7.5 is an example of using a V.R.M.F code to describe the version of IBM WebSphere MQ.

The command level is the command level property of a queue manager; see CommandLevel (MQLONG). Command levels are three-digit codes.

Command levels and versions are related. Up to Version 7.1 the command level and the first three digits of the V.R.M.F. code always matched. From Version 7.1, with the introduction of new-function fix packs, the command level of a queue manager can be greater than the first three digits of an installation. The difference arises, if the queue manager has been associated with a new command level using the **strmqm** command.

From Version 7.1 the rule that links command levels and V.R.M.F levels has been changed. The rule is that when a new version of IBM WebSphere MQ is released, it has a command level greater than released in a new-function fix pack in the previous release. Usually this means that a new release of IBM WebSphere MQ changes the version or release level, rather than the maintenance level.

In Figure 101 on page 705, the maintenance level, on the Y-Axis is labeled with V.R.M.F codes, and the command level, on the X-Axis, with command levels. Note how the illustrative release of `7.R` increases the released command level from `7r0` to `7R0`, and `7R0` exceeds the highest command level shipped in a new-function fix pack, `7r5` .

Reversible and One-way upgrades

The mechanism to apply and remove fix packs varies by platform. You can apply any fix pack that changes only the maintenance or fix level of a release to an installation. Fix pack application is reversible. When you remove a fix pack, you restore the previous release level. So applying `7.r.0.3` to `7.r.0.1`, and then removing it, restores the installation to `7.r.0.1` .

Sometimes, you can change an installation to a particular V.R.M.F level by upgrading the installation with a "manufacturing refresh". If you install a manufacturing refresh, you can only return to the earlier release level by uninstalling, and reinstalling; see "Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342.

Applying a manufacturing refresh to modify the maintenance and fix level of a release is the same process as upgrading to a new version or release of IBM WebSphere MQ. Neither can be reversed without uninstalling.

However there is a particular aspect of upgrading to a new version or release that is different from upgrading to a new maintenance or fix level. If you start a queue manager after a version or release upgrade, the command level of the queue manager is automatically increased. You can then no longer start the queue manager with the installation from the previous release.

On the diagram, an irreversible upgrade is shown by the "One-way" arrow between `7.r` and `7.R`. To prevent an accidental migration, you can rename the new installation. After renaming, rerun the **setmqm** command to associate a queue manager with the new release before running the **strmqm** command to migrate it.

If the upgrade applies only to the maintenance or fix level, you can restart the queue manager with the previous installation, if you reinstall it.

Manufacturing refresh maintenance releases are not distinguished from applying and removing fix packs on the diagram. Both are represented by reversible arrows in Figure 101 on page 705.

Multiple installations

You might choose to have a different installation for each maximum command level supported by an installation. Each column on the diagram would represent a different installation.

You need only one installation at Version 7.1 to be able to select any command level released with Version 7.1 for a queue manager. Eventually, if you intend to run Version 7.1 and version 7.R in parallel, you must have two installations. The scenario that follows uses a single installation.

Another variation is to follow the "rolling fix pack" approach described in "UNIX, Linux, and Windows: Staging maintenance fixes" on page 718. You can maintain two installations at Version 7.1, one at the current fix level, and one either at a later or earlier fix level. You might then install version 7.R as a third installation, or replace the Version 7.1 installation at the older fix level.

Migrating queue managers

The migration paths for queue managers are shown by solid arrows on the diagram. Some of the solid arrows are broken, to avoid cluttering the diagram with too many lines. If the migration to a higher command level jumps command levels, you do not have to migrate it through the intervening commands levels.

To migrate a queue manager to a higher command level in a new-function fix pack, you must start the queue manager with a special parameter:

```
►►──strmqm──┬────────────────────────┬──┬──────────────┬──────────────────────────────────►◄
            └─ -e CMDLEVEL=──Level───┘  └─ QMgrName ───┘
```

Level is the three-digit command level.

The queue manager stops immediately the migration process is complete. When you next start it, it runs at the new command level. The queue manager cannot be restarted at a lower command level. This rule means that you must associate the queue manager with an installation that includes a command level at least as great as the current command level of the queue manager.

Restoring queue managers

To restore a queue manager to a lower command level, you must back up the queue manager before you migrate it to the higher command level.

Procedure

This procedure keeps both QM1 and QM2 at the current maintenance level, QM1 at command level 7r0, and QM2 at the latest command level.

1. Download fix pack 7.r.0.1, when it is released.

   See Fix Central.

   The initial system has two queue managers running 7.r.0.0 at command level 7r0; see Figure 102 on page 708.

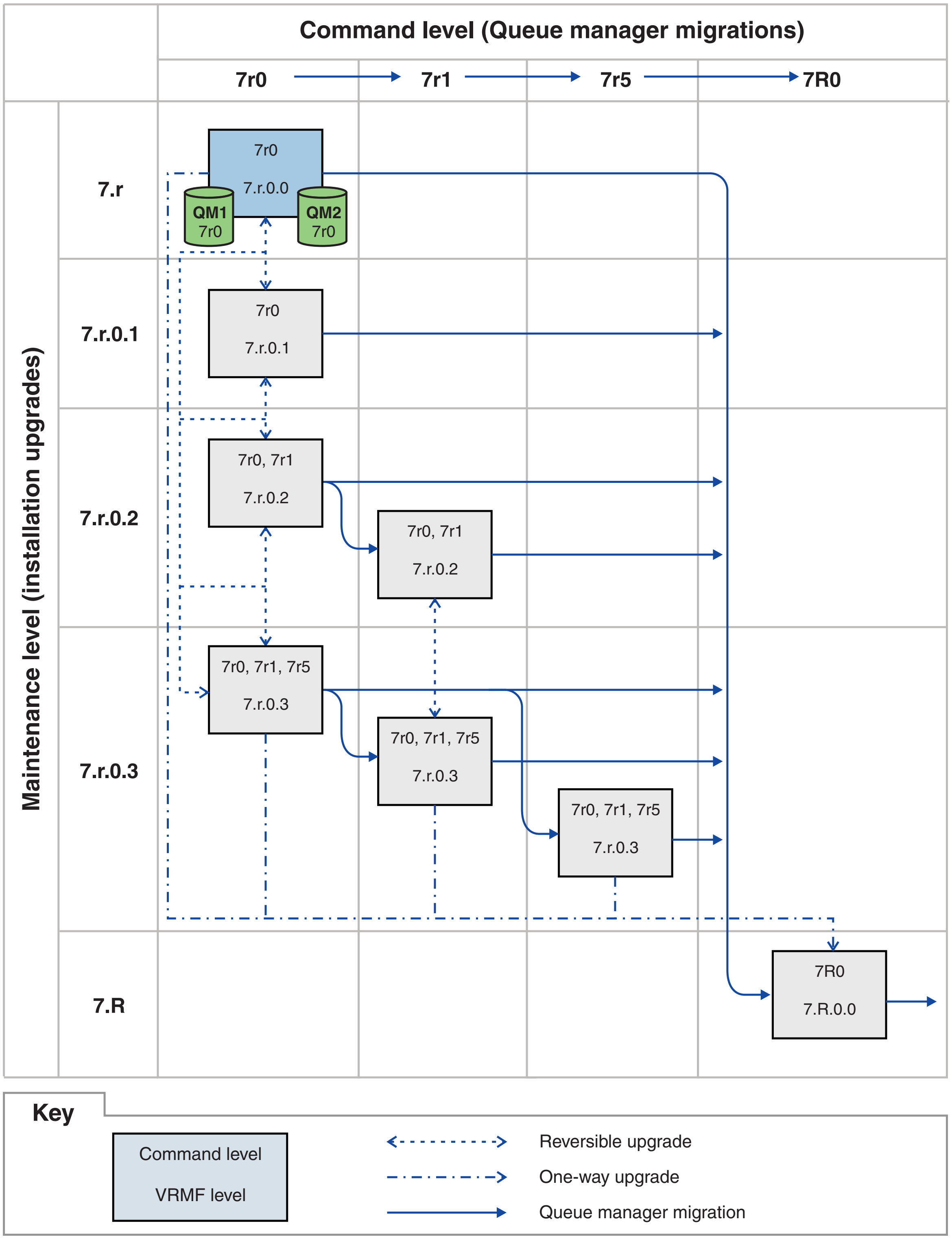


Figure 102. Initial state, `QM1` *and* `QM2` *at command level* `7r0`*, and fix level* `7.r.0.0`

2. Apply fix pack 7.r.0.1 to `Inst_1` .

Follow the task for your platform in “Applying and removing maintenance level updates on platforms other than z/OS” on page 676.

3. Restart the queue managers.

   Both queue managers are now running using `Inst_1` at the `7.r.0.1` maintenance level, and the `7r0` command level; see Figure 103 on page 710 .

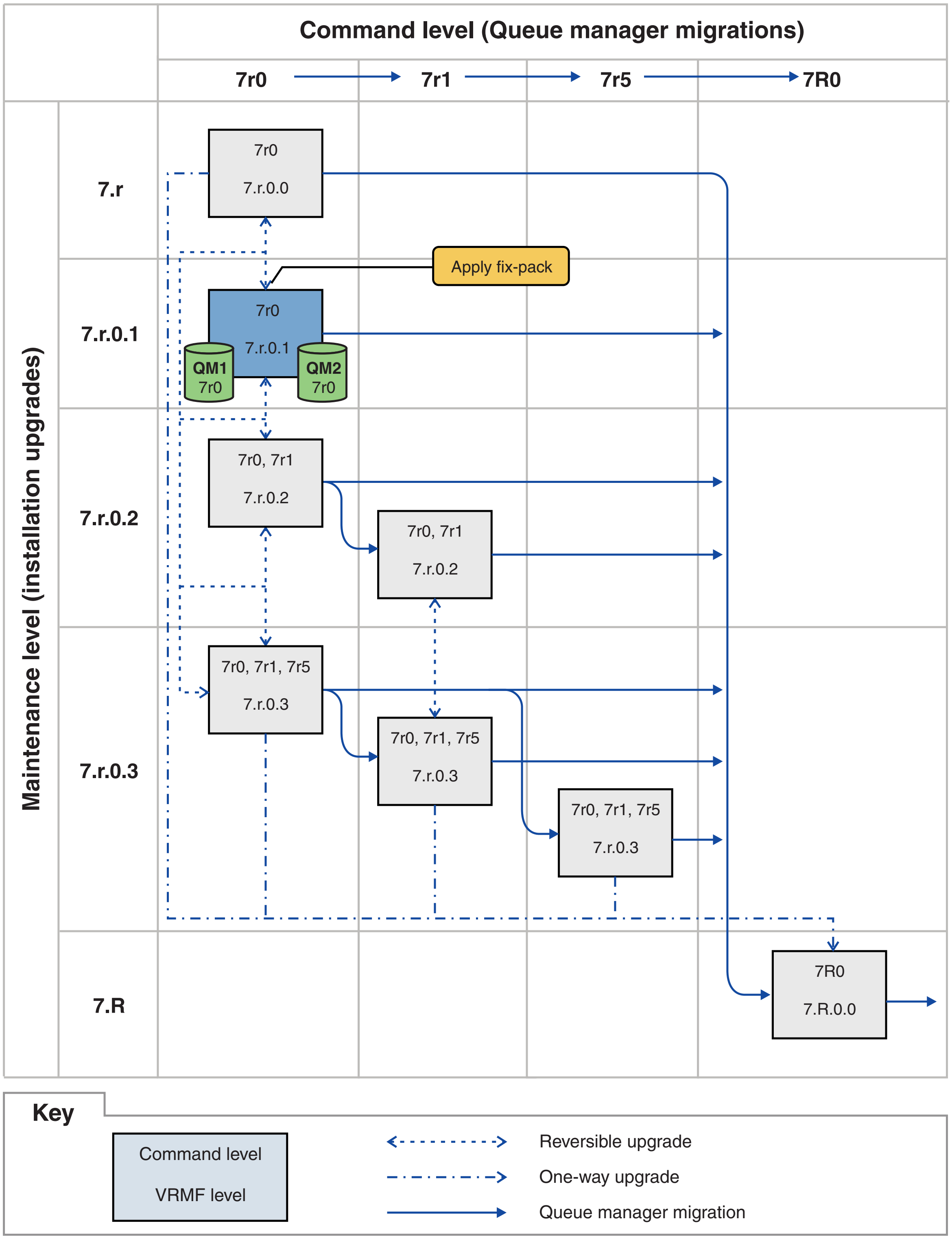


Figure 103. `QM1` and `QM2` at command level `7r0`, and fix level `7.r.0.1`

4. Apply fix pack `7.r.0.2`.
   a. Repeat steps 1 on page 707 and 2 on page 708 with fix pack `7.r.0.2` .

5. Restart QM1.

   QM1 is now running using Inst_1 at the 7.r.0.2 maintenance level, and the 7r0 command level.

   The queue manager is not automatically migrated to the 7r1 command level.

6. Migrate QM2 to the 7r1 command level.

```
strmqm -e CMDLEVEL=711 QM2
```

   QM2 is using Inst_1 at the 7.r.0.2 maintenance level, and has been migrated to the 7r1 command level.

7. Restart QM2.

   QM2 is now running using Inst_1 at the 7.r.0.2 maintenance level, and the 7r1 command level; see Figure 104 on page 712 .

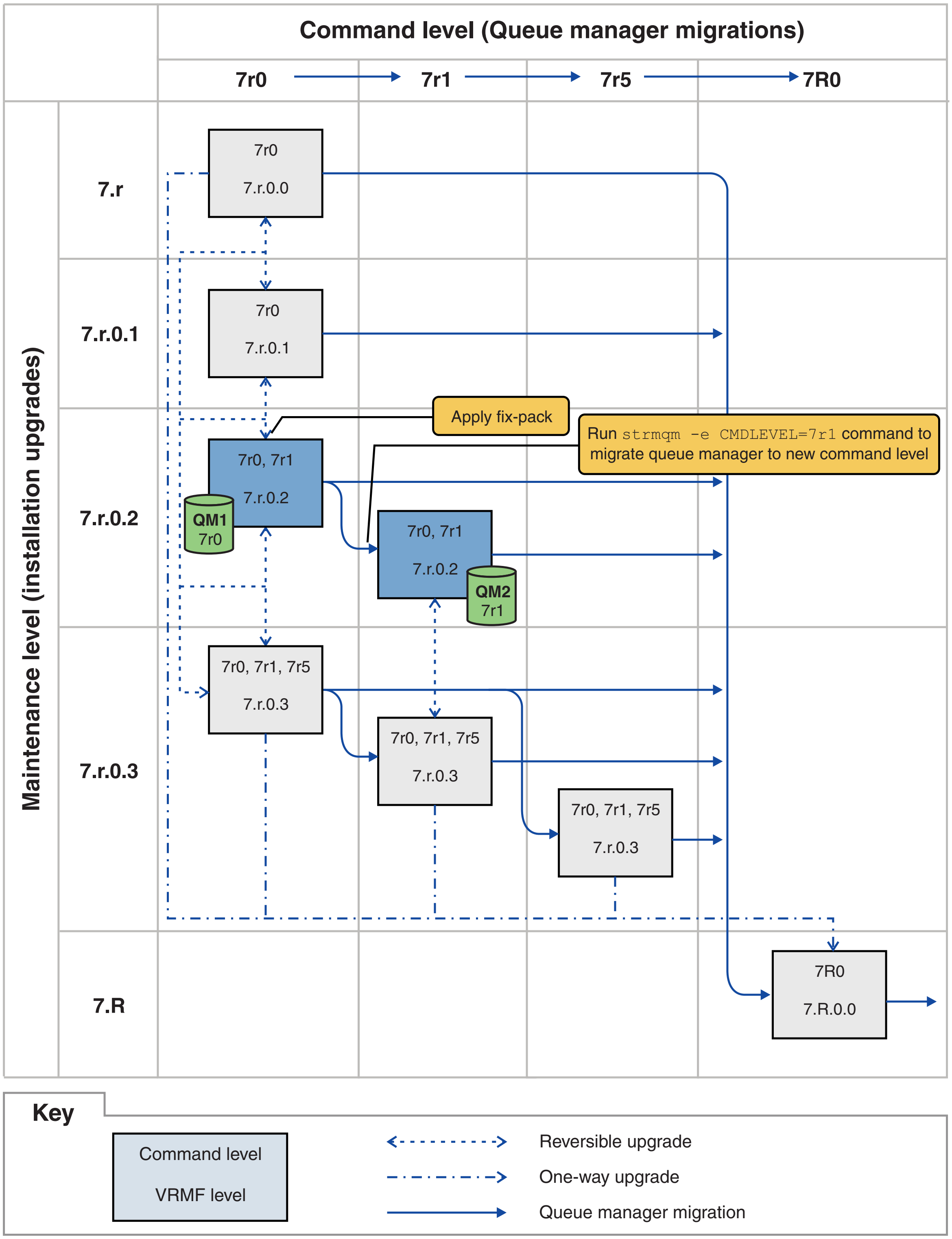


Figure 104. `QM1` at command level `7r0` and fix level `7.r.0.2`; `QM2` at command level `7r1` and fix level `7.r.0.2`

8. Apply fix pack `7.r.0.3` and migrate `QM2` to the `7r5` command level.
   a. Repeat steps 4 on page 710 to 5 on page 711 with fix pack `7.r.0.3` .

b. Repeat steps 6 on page 711 to 7 on page 711 with command level `7r5` .

   `QM1` is using `Inst_1` at the `7.r.0.3` maintenance level, and is running at the `7r0` command level.

   `QM2` is using `Inst_1` at the `7.r.0.3` maintenance level, and has been migrated to the `7r5` command level; see Figure 105 on page 714 .

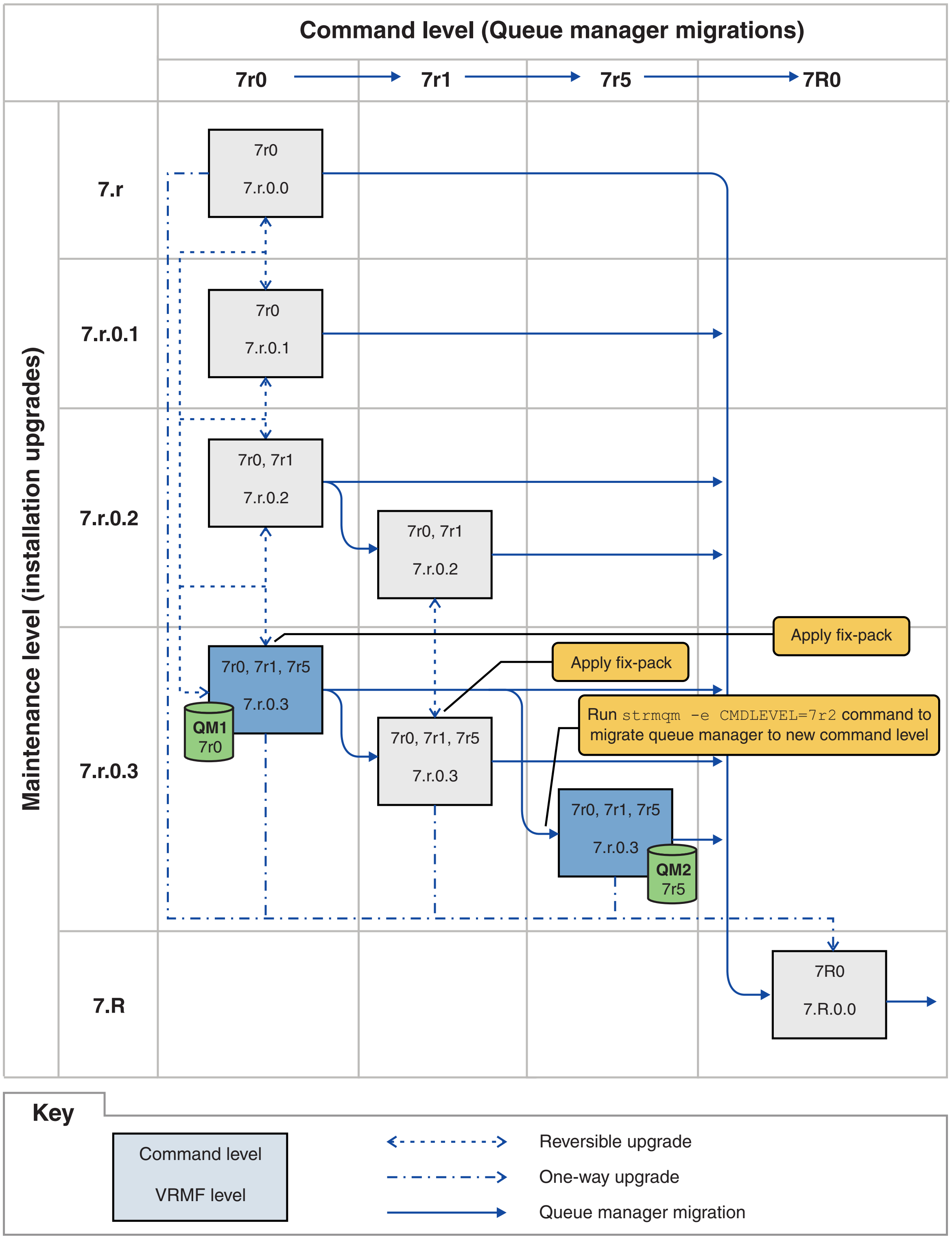


Figure 105. `QM1` at command level `7r0` and fix level `7.r.0.3`; `QM2` at command level `7r5` and fix level `7.r.0.3`

9. Migrate `QM2` to version 7.R
   - On UNIX, Linux, and Windows:

a. Install version 7.R, with the installation name `Inst_2`, alongside Version 7.1 .
b. Set up the local environment to the installation `Inst_2`.

Windows:

```
"Inst_2_INSTALLATION_PATH
\bin\setmqenv" -s
```

The `-s` option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
. Inst_2_INSTALLATION_PATH/bin/setmqenv -s
```

c. Run the **setmqm** command to associate `QM2` with `Inst_2`.

```
setmqm -m QM2 -n Inst_2
```

d. Run the **strmqm** command to start `QM2` and migrate it to version 7.R.

```
strmqm QM2
```

`QM1` is using `Inst_1` at the `7.r.0.3` maintenance level, and is running at the `7r0` command level.

`QM2` is using `Inst_2` at the `7.R.0.0` maintenance level, and has been migrated to the `7R0` command level; see Figure 105 on page 714 .

`Inst_1` remains the primary installation.

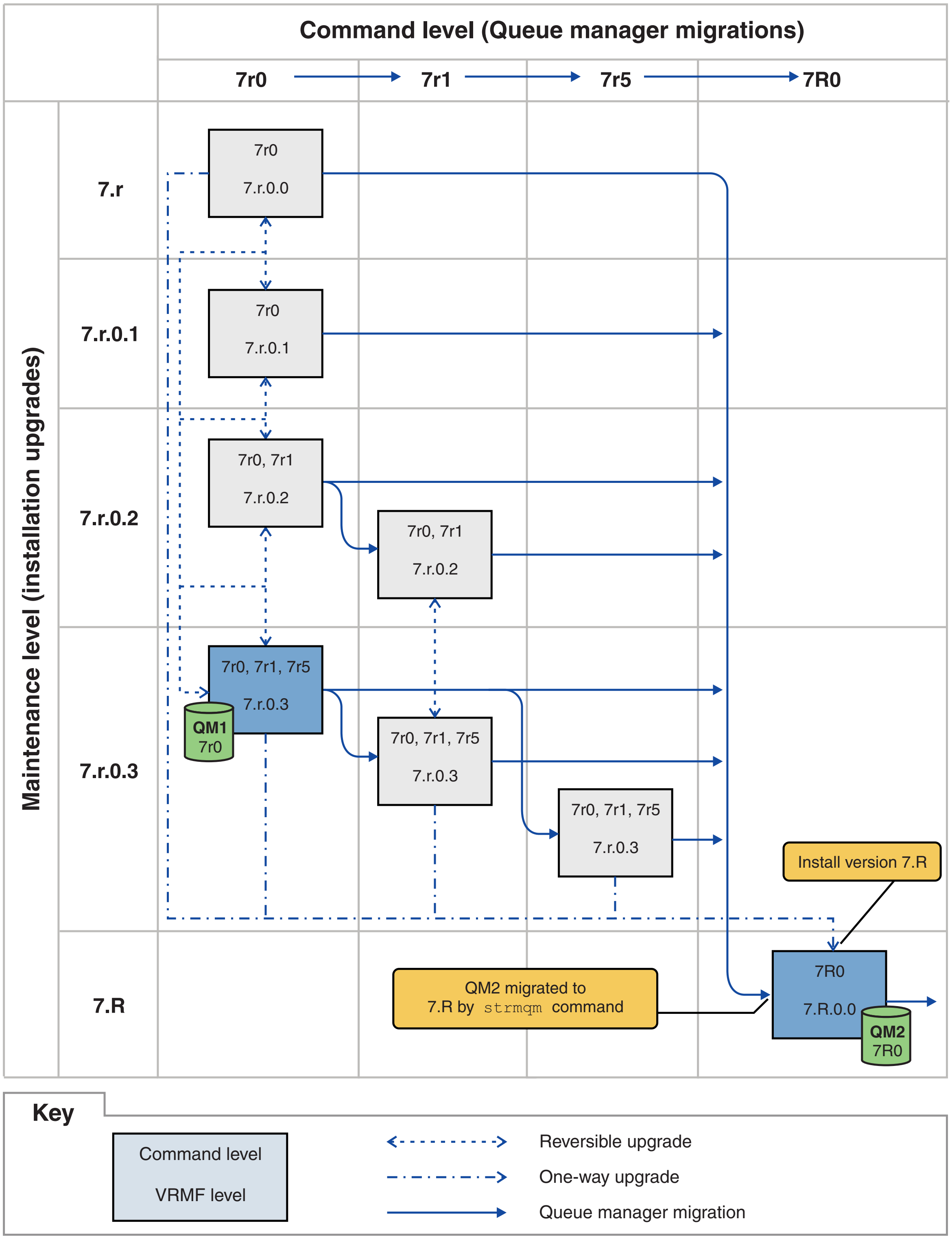


Figure 106. `QM1` at command level `7r0` and fix level `7.r.0.3`; `QM2` at command level `7R0` and fix level `7.R.0.0`

Related concepts:

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

"New function in maintenance level upgrades" on page 341
On platforms other than z/OS, IBM might introduce new functions between releases in maintenance level upgrades such as fix packs. A maintenance level upgrade including new function increases the maximum command level of an installation. When you apply the maintenance level upgrade, the installation supports the new command level. A queue manager associated with the installation is not able to use the new function until you have run **strmqm** to set the queue manager to the new command level.

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"The version naming scheme for WebSphere MQ on platforms other than z/OS" on page 324
On platforms other than z/OS, WebSphere MQ releases have a four-digit Version, Release, Maintenance, and Fix (VRMF) level code. The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

"Upgrade, migration, and maintenance of WebSphere MQ on platforms other than z/OS" on page 342
You can install new releases of WebSphere MQ to upgrade WebSphere MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of WebSphere MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"WebSphere MQ maintenance" on page 336
Maintenance is the application of a reversible fix. The application and identification of maintenance fixes differs on z/OS from other platforms. Any changes to queue manager data are compatible with the previous code level.

"IBM WebSphere MQ migration" on page 338
Migration is the conversion of programs and data to work with a new code level of WebSphere MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

"IBM WebSphere MQ upgrade" on page 337
The term upgrade, used loosely, covers both applying maintenance and modifying the version or release of a product. Upgrading is a change in any of the four V.R.M.F digits. After an upgrade has been applied, as long as no queue managers have been restarted, it can be backed out. The details vary by platform and how the upgrade was applied. Once a queue manager has been restarted, it depends on a number of factors whether the upgrade can be reversed. It depends on the platform, the V.R.M.F level, and the functions and command levels used by the queue manager at the new level.

Related tasks:

"Applying and removing maintenance level updates on platforms other than z/OS" on page 676
When you apply and remove maintenance level updates to IBM WebSphere MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix. The following topics describe how to apply and remove fix packs on platforms other than z/OS

"UNIX, Linux, and Windows: Staging maintenance fixes" on page 718
Use multiple installations of IBM WebSphere MQ on the same server to control the release of maintenance fixes.

Related information:

CommandLevel (MQLONG)

Querying the maintenance level

Query the IBM WebSphere MQ maintenance level by running the **dspmqver** command

About this task

After an update to the initial installation, the version indicates the maintenance level to which the product has been updated. For example, before applying any maintenance, the version is 7.0.1.0. As maintenance is applied the last two digits will be updated, for example to 7.0.1.3.

Procedure

To view the version use the `dspmqver` command. At a command prompt, enter the following command: `dspmqver`. The resulting messages include the IBM WebSphere MQ version number, which shows the maintenance level.

Related information:

dspmqver

UNIX, Linux, and Windows: Staging maintenance fixes

Use multiple installations of IBM WebSphere MQ on the same server to control the release of maintenance fixes.

Before you begin

Set up your configuration modelled on the first row of Figure 107 on page 719. You can apply this scenario to any version of IBM WebSphere MQ fromIBM WebSphere MQ Version 7.1 onwards. In this scenario it is assumed you have a number of applications and two queue managers, `QM1` and `QM2`, running on a server. IBM WebSphere MQ Version 7.0.1 is not installed on the server.

1. Install two copies of IBM WebSphere MQ Version 7.1. In the example, they are named `Inst_1` and `Inst_2`.
2. Make `Inst_1` primary by running **setmqinst**.
3. Associate all the queue managers on the server with `Inst_1` by running **setmqm**.
4. Start all the queue managers on the server.
5. Show and connect all direct connections with the queue managers associated with `Inst_1` in IBM WebSphere MQ Explorer.
6. Set up remote connections to all the queue managers in each instance of IBM WebSphere MQ Explorer.

About this task

You can install multiple copies of IBM WebSphere MQ on a server to stage the release of IBM WebSphere MQ fixes. Figure 107 on page 719 illustrates a way of using two installations to roll out fixes. In this approach, you maintain two fix levels on a server, with the aim of getting all queue managers and applications to the production fix level before replacing the previous level on fix pack with the next level.

Which installation an application uses is driven by the queue manager an application connects to. The **setmqm** command associates a queue manager with an installation. You can associate a queue manager with a different installation as long as the installation is at the same or higher command level. In this example, all the installations are at the same command level. You can associate or reassociate a queue manager with either of the installations running any of the fix packs.

In the example, an application links to the primary installation. When it connects to a queue manager, IBM WebSphere MQ switches the linkage to the installation associated with the queue manager; see "Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348.

For applications built with the link options described in the product documentation, the simplest way to configure the link library search path for IBM WebSphere MQ applications is to make an installation primary. Only if it is important to pick up a fix in the IBM WebSphere MQ link library itself, must you review the search path. Either you must make the installation with the IBM WebSphere MQ link library fix primary, or make a local adjustment for the application, perhaps by running the **setmqenv** command.

Running commands is a different matter. Commands are always run from the primary installation, or the installation you have selected by running the **setmqenv** command. If you run a command from the wrong installation, the command fails. For example, if `QM1` is associated with `Inst_1`, running the Windows command, `Inst_2_Installation_path/bin/strmqm QM1` fails.

If you are using IBM WebSphere MQ Explorer and you have two installations, you also have two IBM WebSphere MQ Explorer instances. One linked to one installation, and one to the other. Each IBM WebSphere MQ Explorer shows locally connected queue managers that are associated with the same installation as the instance of IBM WebSphere MQ Explorer. To monitor all the queue managers on a server, set up remote connections to the queue managers associated with the other installations.

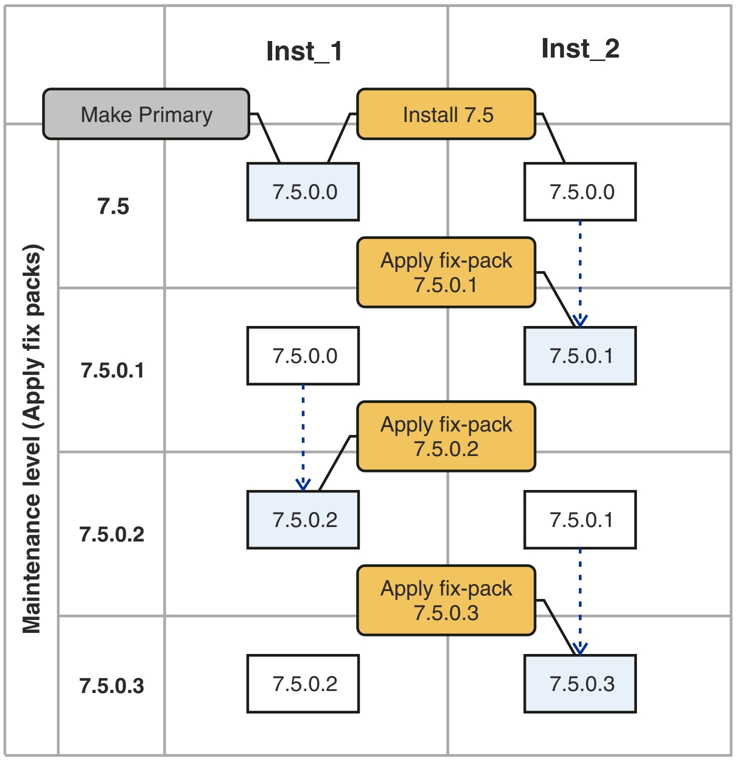


Figure 107. Rolling fix releases

Procedure

1. Download fix pack 7.1.0.1, when it is released.

   See Fix Central.

2. Apply fix pack 7.1.0.1 to `Inst_2`.

Follow the task for your platform in "Applying and removing maintenance level updates on platforms other than z/OS" on page 676.

3. Verify Inst_2.
4. Transfer the queue managers to Inst_2 one at a time.
   a. Stop QM1 and the applications connected to it.

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.
   b. Set up the local environment to the installation Inst_2.

Windows:

```
"Inst_2_INSTALLATION_PATH\bin\setmqenv" -s
```

The -s option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
. Inst_2_INSTALLATION_PATH/bin/setmqenv -s
```

   c. Associate the queue manager with Inst_2.

```
setmqm -m QM1 -n Inst_2
```

   d. Start QM1

```
strmqm QM1
```

   e. Repeat substeps c and d for QM2.
   f. Set up IBM WebSphere MQ Explorer for Inst_2.
      1) Start the Inst_2 instance of IBM WebSphere MQ Explorer

         Tip: On Windows, hover over the IBM WebSphere MQ icon in the system tray. The hover help shows the installation name associated with the IBM WebSphere MQ Explorer instance.
      2) Click **IBM WebSphere MQ** > **Queue Managers** > **Show/Hide Queue Managers...** >
      3) Click each directly connected queue manager listed in the Hidden Queue Managers list > **Show**.
      4) Click **Close**.
5. Set Inst_2 primary.

Windows:

```
"Inst_2_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_2
```

UNIX:

```
Inst_2_INSTALLATION_PATH/bin/setmqinst -i -n Inst_2
```

6. Down load fix pack 7.1.0.2, when it is released.

   See Fix Central.
7. Apply fix pack 7.1.0.2 to Inst_1.

   Follow the task for your platform in "Applying and removing maintenance level updates on platforms other than z/OS" on page 676.
8. Verify Inst_1.
9. Transfer queue managers to Inst_1 one at a time.
   a. Follow the procedure in step 4

      Replacing Inst_2 by Inst_1 in the instructions.

10. Set Inst_1 primary.

Windows:

```
"Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
```

UNIX:

```
Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
```

11. Repeat steps 1 on page 719 to 5 on page 720 for 7.1.0.3, 7.1.0.5, and so on
12. Repeat steps 6 on page 720 to 10 for 7.1.0.4, 7.1.0.6, and so on.

Related concepts:

"Queue manager coexistence in Version 7.5" on page 345
Queue managers, with different names, can coexist on any server as long as they use the same IBM WebSphere MQ installation. On UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 348
You can install multiple copies of IBM WebSphere MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks:

"Migrating IBM WebSphere MQ library loading from Version 7.0.1 to Version 7.5" on page 483
No change in the way IBM WebSphere MQ libraries are loaded is normally required if you upgrade from Version 7.0.1 to Version 7.5. You must have followed the instructions on building IBM WebSphere MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ Version 7.0.1 with IBM WebSphere MQ Version 7.5. If you choose to take advantage of multi-installation in IBM WebSphere MQ Version 7.5, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM WebSphere MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Related information:

Installing WebSphere MQ server on Windows
This topic describes how to install IBM WebSphere MQ server on Windows systems by using the Launchpad. This procedure can be used for installing a first or a subsequent installation.

Associating a queue manager with an installation
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

Changing the primary installation
You can use the **setmqinst** command to set or unset an installation as the primary installation.

setmqenv

setmqinst

setmqm

Migration commands, utilities, and reference information

A selection of commands, utilities, and application reference information related to migration are collected together in the following subtopics.

JMS PROVIDERVERSION property

The JMS **PROVIDERVERSION** property selects whether a Java application publishes and subscribes using the queued command message interface, or the integrated call interface.

Rules for selecting the WebSphere MQ messaging provider mode

The WebSphere MQ messaging provider has two modes of operation: WebSphere MQ messaging provider normal mode, and WebSphere MQ messaging provider migration mode. You can select which mode a JMS application uses to publish and subscribe.

The WebSphere MQ messaging provider normal mode uses all the features of a MQ queue manager to implement JMS. This mode is used only to connect to a WebSphere MQ queue manager and can connect to queue managers in either client or bindings mode. This mode is optimized to use the new function.

If you are not using WebSphere MQ Real-Time Transport, the mode of operation used is determined primarily by the **PROVIDERVERSION** property of the connection factory. If you cannot change the connection factory that you are using, you can use the `com.ibm.msg.client.wmq.overrideProviderVersion` property to override any setting on the connection factory. This override applies to all connection factories in the JVM but the actual connection factory objects are not modified.

You can set **PROVIDERVERSION** to the possible values: `7`, `6`, or *unspecified*. However, **PROVIDERVERSION** can be a string in any one of the following formats:

- V.R.M.F
- V.R.M
- V.R
- V

where V, R, M and F are integer values greater than or equal to zero.

7 - Normal mode

Uses the WebSphere MQ messaging provider normal mode.

If you set **PROVIDERVERSION** to 8 only the WebSphere MQ messaging provider normal mode of operation is available. If the queue manager specified in the connection factory settings is not a Version 7.0.1, or later, queue manager, the createConnection method fails with an exception.

The WebSphere MQ messaging provider normal mode uses the sharing conversations feature and the number of conversations that can be shared is controlled by the **SHARECNV()** property on the server connection channel. If this property is set to `0`, you cannot use WebSphere MQ messaging provider normal mode and the createConnection method fails with an exception.

6 - Migration mode

Uses the WebSphere MQ messaging provider migration mode.

The WebSphere MQ classes for JMS use the features and algorithms supplied with WebSphere MQ version 6.0. If you want to connect to WebSphere Event Broker version 6.0 or WebSphere Message Broker version 6.0 or 6.1 using WebSphere MQ Enterprise Transport version 6.0, you must use this mode. You can connect to a WebSphere MQ version 7.0.1 queue manager using this mode, but none of the new features of a version 7.0.1 queue manager are used, for example, read ahead or streaming. If you have a WebSphere MQ version 7.0.1 client connecting to a WebSphere MQ version 7.0.1 queue manager on a distributed platform or a WebSphere MQ version 7.0.1 queue manager on z/OS, then the message selection is done by the queue manager rather than on the client system.

unspecified

The **PROVIDERVERSION** property is set to *unspecified* by default.

A connection factory that was created with a previous version of WebSphere MQ classes for JMS in JNDI takes this value when the connection factory is used with the new version of WebSphere MQ classes for JMS. The following algorithm is used to determine which mode of operation is used. This algorithm is used when the createConnection method is called and uses other aspects of the connection factory to determine if WebSphere MQ messaging provider normal mode or WebSphere MQ messaging provider migration mode is required.

1. First, an attempt to use WebSphere MQ messaging provider normal mode is made.
2. If the queue manager connected is not WebSphere MQ version 7.0.1, or later, the connection is closed and WebSphere MQ messaging provider migration mode is used instead.
3. If the **SHARECNV** property on the server connection channel is set to 0, the connection is closed and WebSphere MQ messaging provider migration mode is used instead.
4. If **BROKERVER** is set to V1 or the new default *unspecified* value, WebSphere MQ messaging provider normal mode continues to be used, and therefore any publish/subscribe operations use the new WebSphere MQ version 7.0.1, or later, features.

   See ALTER QMGR for information about the PSMODE parameter of the ALTER QMGR command for further information on compatibility.
5. If **BROKERVER** is set to V2 the action taken depends on the value of **BROKERQMGR** :
   - If the **BROKERQMGR** is blank:

     If the queue specified by the **BROKERCONQ** property can be opened for output (that is, MQOPEN for output succeeds) and **PSMODE** on the queue manager is set to COMPAT or DISABLED, then WebSphere MQ messaging provider migration mode is used.
   - If the queue specified by the **BROKERCONQ** property cannot be opened for output, or the **PSMODE** attribute is set to ENABLED:

     WebSphere MQ messaging provider normal mode is used.
   - If **BROKERQMGR** is non-blank :

     WebSphere MQ messaging provider mode is used.

Related information:

When to use PROVIDERVERSION
In two cases you must override the default selection of **PROVIDERVERSION** for the WebSphere MQ classes for JMS to work correctly.

BROKERQMGR

BROKERCONQ

PSMODE

When to use PROVIDERVERSION

In two cases you must override the default selection of **PROVIDERVERSION** for the WebSphere MQ classes for JMS to work correctly.

There are two scenarios where you cannot use the algorithm described in Rules for selecting the WebSphere MQ messaging provider mode; consider using **PROVIDERVERSION** in these scenarios.

1. If WebSphere Event Broker or WebSphere Message Broker is in compatibility mode, you must specify **PROVIDERVERSION** for them to work correctly.
2. If you are using WebSphere Application Server Version 6.0.1, WebSphere Application Server Version 6.0.2, or WebSphere Application Server Version 6.1, connection factories are defined using the WebSphere Application Server administrative console.

   In WebSphere Application Server the default value of the **BROKERVER** property on a connection factory is V2. The default **BROKERVER** property for connection factories created by using **JMSAdmin** or WebSphere MQ Explorer is V1. This property is now “unspecified” in WebSphere MQ.

If **BROKERVER** is set to V2 (either because it was created by WebSphere Application Server or the connection factory has been used for publish/subscribe before) and has an existing queue manager that has a **BROKERCONQ** defined (because it has been used for publish/subscribe messaging before), the WebSphere MQ messaging provider migration mode is used.

However, if you want the application to use peer-to-peer communication and the application is using an existing queue manager that has ever done publish/subscribe, and has a connection factory with **BROKERVER** set to 2 (if the connection factory was created in WebSphere Application Server this is the default), the WebSphere MQ messaging provider migration mode is used. Using WebSphere MQ messaging provider migration mode in this case is unnecessary; use WebSphere MQ messaging provider normal mode instead. You can use one of the following methods to work around this:

- Set **BROKERVER** to 1 or unspecified. This is dependent on your application.
- Set **PROVIDERVERSION** to 7, which is a custom property in WebSphere Application Server Version 6.1. The option to set custom properties in WebSphere Application Server Version 6.1 and later is not currently documented in the WebSphere Application Server product documentation.

  Alternatively, use the client configuration property, or modify the queue manager connected so it does not have the **BROKERCONQ**, or make the queue unusable.

strmqbrk: Migrate the IBM WebSphere MQ Version 6.0 publish/subscribe broker to Version 7.5

Migrate the persistent state of a IBM WebSphere MQ publish/subscribe broker to a Version 7.5 queue manager.

Purpose

Use the **strmqbrk** command to migrate the state of a IBM WebSphere MQ Version 6.0 publish/subscribe broker to IBM WebSphere MQ Version 7.5 queue manager. If the queue manager has already been migrated, no action is taken.

In IBM WebSphere MQ Version 6.0, **strmqbrk** started a broker. IBM WebSphere MQ Version 7.5 publish/subscribe cannot be started in this manner. To enable publish/subscribe for a queue manager, use the **ALTER QMGR** command.

You can also use the **runmqbrk** command. This has the same parameters as **strmqbrk** and exactly the same effect.

Syntax

AIX, HP-UX, Linux, Solaris, and Windows

```
                                    ┌─Default queue manager─┐
►►──strmqbrk──┬────────────────────┬─┼───────────────────────┼─┬──────┬──────────────────►
              └─ -p ──ParentQMgrName─┘ └─ -m ──QMgrName────────┘ └─ -f ─┘

   ┌─QMgrDataDir──/──psmigr.log─┐
►──┼──────────────────────────┼──────────────────────────────────────────────────────►◄
   └─ -l ──LogFileName────────┘
```

Optional parameters

AIX, HP-UX, Linux, Solaris, and Windows

-p *ParentQMgrName*

> **Note:** This option is deprecated. **strmqbrk** migrates the parent connection automatically.
> If you specify the current parent queue manager, a warning message is issued and migration continues. If you specify a different queue manager, a error is issued and migration is not performed.

-m *QMgrName*
The name of the queue manager to be migrated. If you do not specify this parameter, the command is routed to the default queue manager.

-f Force migration. This option specifies that objects created during the migration replace existing objects with the same name. If this option is not specified, if migration would create a duplicate object, a warning is issued, the object is not created, and migration continues.

-l *LogFileName*
Log migration activity to the file specified in *LogFileName*.

Syntax

IBM i

```
►►──STRMQMBRK──┬──────────────────────────────────┬─┬────────────────────┬──────────────────────────►◄
               └─ -PARENTMQM──(ParentQMgrName)────┘ └─ -MQMNAME ──QMgrName─┘
```

Optional parameters

AIX, HP-UX, Linux, Solaris, and Windows

-PARENTMQM*(ParentQMgrName)*

Note: This option is deprecated.
If you specify the current parent queue manager, a warning message is issued and migration continues. If you specify a different queue manager, a warning is issued and migration is not performed.

-MQMNAME *QMgrName*
The name of the queue manager to be migrated. If you do not specify this parameter, the command is routed to the default queue manager.

Related information:
ALTER QMGR

migmbbrk

The migmbbrk command migrates publish/subscribe configuration data from WebSphere Event Broker Version 6.0 or WebSphere Message Broker Version 6.0 or 6.1 to WebSphere MQ Version 7.0.1 or later versions.

Purpose

The migmbbrk command is not supported on all of the platforms that WebSphere MQ supports. See *Supported operating systems* **for details.**

To use the **migmbbrk** command you must be using at least WebSphere Message Broker Version 6.0, Fix Pack 9, or WebSphere Message Broker Version 6.1, Fix Pack 4.

Use the **migmbbrk** command to migrate the publish/subscribe configuration data from a WebSphere Event Broker Version 6.0 or a WebSphere Message Broker Version 6.0 or Version 6.1 broker to a WebSphere MQ Version 7.0.1 or later queue manager. The command runs a migration process that migrates the following publish/subscribe configuration data to the queue manager that is associated with the named broker:

- Subscriptions
- Subscription points. (Subscription points are supported only when RFH2 messages are used.)
- Streams
- Retained publications

The **migmbbrk** command does not migrate the Access Control List (ACL). Instead, running the migration with the -t or -r parameters produces a file containing suggested setmqaut commands to set up a security environment in the queue manager that is equivalent to the security environment that existed in the broker. You must review and modify the security command file as needed and run the commands to set up a security environment in the queue manager, equivalent to the one that existed in the broker, before you run the migration with the -c parameter to complete the migration.

Note: On UNIX systems, all authorities are held by user groups internally, not by principals. This has the following implications:

- If you use the **setmqaut** command to grant an authority to a principal, the authority is granted to the primary user group of the principal. This means that the authority is effectively granted to all members of that user group.
- If you use the **setmqaut** command to revoke an authority from a principal, the authority is revoked from the primary user group of the principal. This means that the authority is effectively revoked from all members of that user group.

You must issue the **migmbbrk** command from a command window that can execute both WebSphere MQ and WebSphere Message Broker commands successfully. Typically this is true if the command is issued from a WebSphere Message Broker command console.

The WebSphere Event Broker Version 6.0 or WebSphere Message Broker Version 6.0 or 6.1 publish/subscribe configuration data, which is stored in the subscription database tables, is not deleted by the migration process. This configuration data is therefore available to use until you explicitly delete it.

Syntax

```
                           ┌───────────────┐
                           ▼               │
►►──migmbbrk──┬─ -c ─┬───────┬─────────┬───┴── -b ──BrokerName──────────────────────────────►◄
              ├─ -r ─┤       ├─ -l ─┤
              └─ -t ─┘       ├─ -o ─┤
                             ├─ -s ─┤
                             └─ -z ─┘
```

Required parameters

-b *BrokerName*
The name of the broker that is the source of the publish/subscribe configuration data that is to be migrated. The queue manager to which the publish/subscribe configuration data is migrated is the queue manager that is associated with the named broker.

-c
Complete the migration of the publish/subscribe configuration data. The completion phase of the migration uses the topic objects that are created in the initial -t phase. It is possible that the broker state has changed since the initial phase was run and that new additional topic objects are now required. If so, the completion phase creates new topic objects as necessary. The completion phase does not delete any topic objects that have become unnecessary; you might need to delete any topic objects that you do not require.

Before you complete the migration you must review and modify the security command file produced in the -r or -t phase as required and execute the commands to set up a security environment in the queue manager, equivalent to the one that existed in the broker.

Before you run this completion phase, you must run the initial -t phase. You cannot use the -c parameter with the -r parameter or the -t parameter. This phase also creates a migration log.

-r

Rehearse the migration process but do not change anything. You can use this parameter before running the migration with the -t parameter, to create a migration log, including any errors, so that you can observe what the result of the migration process would be, but without changing the current configurations.

Rehearsing the migration also produces a file containing suggested setmqaut commands to set up a security environment in the queue manager that is equivalent to the security environment that existed in the broker. Before you complete the migration with the -c parameter you must review and modify the security command file as required and execute the commands to set up a security environment in the queue manager, equivalent to the one that existed in the broker.

You cannot use the -r parameter with the -c parameter or the -t parameter.

-t

Create topic objects that might be needed in the queue manager, based on the ACL entries that are defined in the broker.

Use of the -t parameter also produces a file containing suggested setmqaut commands to set up a security environment in the queue manager that is equivalent to the security environment that existed in the broker. The topic objects are created in anticipation of you executing the security commands to create ACLs for the topic objects. Before you complete the migration with the -c parameter you must review and modify the security command file as required and execute the commands to set up a security environment in the queue manager, equivalent to the one that existed in the broker.

You must run this phase before you run the completion phase with the -c parameter. You cannot use the -t parameter with the -c parameter or the -r parameter. This phase also creates a migration log.

Optional parameters

-l

Leave the broker running. If you do not specify this parameter, the broker is shut down by default at the end of the migration process.

-o

Overwrite any subscription or retained publication that exists in the queue manager and that has the same name as a subscription or retained publication that is being migrated from the broker, with the publish/subscribe configuration data that was retrieved from the broker. The -o parameter has no effect if you use it with the -r parameter.

-s

Discard any intermediate configuration data that was retained from a previous instance of the migration process that failed or was interrupted. The migration process populates private queues with temporary data. If the migration process completes successfully, the temporary data is deleted. If you do not specify this parameter and the migration process fails or is interrupted, the temporary data is retained and is used by the migration process if you restart it, so that the process resumes at the point where it previously failed or was interrupted.

-z

Run the migration process, regardless of whether it has previously run to a successful completion. If you do not specify this parameter and the migration process has previously run to a successful completion, the process recognizes this fact and exits. You can use the -o parameter with the -z parameter, but this is not mandatory. A previous rehearsal of the migration using the -r parameter does not count as a successful completion.

Return codes

Return Code	Explanation
0	Migration completed successfully
20	An error occurred during processing

Output files

The migration process writes two output files to the current directory:

amqmigrateacl.txt
A file containing a list of setmqaut commands, created in the current directory for you to review, change, and run if appropriate, to help you to reproduce your ACLs.

amqmigmbbrk.log
A log file containing a record of the details of the migration.

Examples

This command migrates the publish/subscribe configuration data of broker BRK1 into its associated queue manager and specifies that the migration process runs regardless of whether it has previously run to a successful completion. It also specifies that any subscription or retained publication that exists in the queue manager, that has the same name as a subscription or retained publication that is being migrated from the broker, must be overwritten.

```
migmbbrk -z -o -b BRK1
```

Supported operating systems

The **migmbbrk** command is supported only on the following platforms that support WebSphere Event Broker Version 6.0 or WebSphere Message Broker Version 6.0:

Microsoft Windows XP Professional with SP2, 32-bit versions only
Solaris x86-64 platform: Solaris 10
Solaris SPARC platform: Sun Solaris 9 (64-bit)
AIX Version 5.2 or later, 64-bit only
HP-UX Itanium platform: HP-UX 11i
Linux zSeries (64-bit)
Linux PowerPC® (64-bit)
Linux Intel x86
Linux Intel x86-64

On z/OS, the equivalent function to the migmbbrk command is provided by the CSQUMGMB utility.

Publish/subscribe migration log file

The publish/subscribe migration log file is called `amqmigmbbrk.log`. It shows the details of the migration.

Identifying subscriptions in the log file or error messages

The log file identifies a subscription in the log file or an error message by listing its properties, including its topic string. The topic string can be up to 10,240 characters long, and is expressed in Unicode in the broker. The topic string might be too long or contain characters that cannot be represented in the current character set.

To identify subscriptions precisely, the migration process assigns a sequence number to every broker subscription that it encounters. The sequence starts at 1. When a subscription is identified by the migration process, the identification includes the sequence number, and anything else that is known about the subscription.

You can use the **mqsireportproperties** broker command to list subscriptions in the broker in the same order as the migration process. For example, the fourth subscription that the migration process counts is also the fourth that is listed by **mqsireportproperties**. With this information, you can correlate the sequence number from the log file or error message with the subscription in the broker.

The broker command to list the subscriptions is as follows:

```
mqsireportproperties brokername -e default -o DynamicSubscriptionEngine -r
```

To store the results in a file called, for example, `outfile.txt`, the command is as follows:

```
mqsireportproperties brokername -e default -o DynamicSubscriptionEngine -r > outfile.txt
```

Example

The following log file lists the actions that have been performed by a successful migration process. It shows the migration of subscriptions, ACLs, and retained publications.

```
2009-01-28 11:43:54.187: Migrating Pub/Sub state from Websphere Message Broker: TEST_BROKER
2009-01-28 11:43:54.187: Into queue manager: TEST_QM
2009-01-28 11:43:54.187: Command switches:
2009-01-28 11:43:54.187:  -t
2009-01-28 11:43:54.187:  -z
2009-01-28 11:43:54.187:  -l
2009-01-28 11:43:55.484: Starting to parse subscriptions ...
2009-01-28 11:43:55.484: Migrating subscriptions for topic string RFH2/EU/France/Toison
2009-01-28 11:43:55.484: [1] Migrating subscription for:
2009-01-28 11:43:55.484:  Format: mqrfh2
2009-01-28 11:43:55.484:  Queue Manager: PSMIG_QM
2009-01-28 11:43:55.484:  Queue: PUBSUB.FRANCE.QUEUE
2009-01-28 11:46:23.968: Migrating subscriptions for topic string RFH2/EU/France/Carnac
2009-01-28 11:46:23.968: [2] Migrating subscription for:
2009-01-28 11:46:23.968:  Format: mqrfh2
2009-01-28 11:46:23.968:  Queue Manager: PSMIG_QM
2009-01-28 11:46:23.968:  Queue: PUBSUB.FRANCE.QUEUE
2009-01-28 11:46:23.968: Migrating subscriptions for topic string $SYS/STREAM/TEST_STREAM/RFH1/EU/France/Pontivy
2009-01-28 11:46:23.984: [3] Migrating subscription for:
2009-01-28 11:46:23.984:  Format: mqrfh2
2009-01-28 11:46:23.984:  Queue Manager: PSMIG_QM
2009-01-28 11:46:23.984:  Queue: PUBSUB.FRANCE.QUEUE
2009-01-28 11:46:24.031: Migrating subscriptions for topic string $SYS/Broker/+/warning/expiry/Subscription/#
2009-01-28 11:46:24.031: [4] Migrating subscription for:
2009-01-28 11:46:24.031:  Format: mqrfh2
2009-01-28 11:46:24.031:  Queue Manager: PSMIG_QM
2009-01-28 11:46:24.031:  Queue: PUBSUB.SAMPLE.QUEUE
2009-01-28 11:46:24.125: Migrating subscriptions for topic string $SYS/Broker/+/Subscription/#
2009-01-28 11:46:24.125: [5] Migrating subscription for:
2009-01-28 11:46:24.125:  Format: mqrfh2
2009-01-28 11:46:24.125:  Queue Manager: PSMIG_QM
2009-01-28 11:46:24.125:  Queue: PUBSUB.SAMPLE.QUEUE
2009-01-28 11:46:24.140: Migrating subscriptions for topic string $SYS/Broker/+/Status
2009-01-28 11:46:24.140: [6] Migrating subscription for:
2009-01-28 11:46:24.140:  Format: mqrfh2
2009-01-28 11:46:24.140:  Queue Manager: PSMIG_QM
2009-01-28 11:46:24.140:  Queue: PUBSUB.SAMPLE.QUEUE
2009-01-28 11:46:24.156: Migrating subscriptions for topic string $SYS/Broker/+/Status/ExecutionGroup/#
2009-01-28 11:46:24.156: [7] Migrating subscription for:
2009-01-28 11:46:24.156:  Format: mqrfh2
2009-01-28 11:46:24.156:  Queue Manager: PSMIG_QM
2009-01-28 11:46:24.156:  Queue: PUBSUB.SAMPLE.QUEUE
2009-01-28 11:46:24.250: Migrating subscriptions for topic string $SYS/STREAM/TEST_STREAM/RFH1/EU/France/Kersaux
2009-01-28 11:46:24.250: [8] Migrating subscription for:
2009-01-28 11:46:24.250:  Format: mqrfh2
2009-01-28 11:46:24.250:  Queue Manager: PSMIG_QM
2009-01-28 11:46:24.250:  Queue: PUBSUB.FRANCE.QUEUE
2009-01-28 11:46:24.281: ... finished parsing subscriptions
```

```
2009-01-28 11:46:24.281: Starting to parse topics ...
2009-01-28 11:46:24.281: Migrating ACLs for topic string
2009-01-28 11:46:24.281: Migrating ACLs for topic string RFH2/EU/France/Toison
2009-01-28 11:46:24.281: Migrating ACLs for topic string RFH2/EU/France/Carnac
2009-01-28 11:46:24.281: Migrating ACLs for topic string $SYS/STREAM/TEST_STREAM/RFH1/EU/France/Pontivy
2009-01-28 11:46:24.281: Migrating ACLs for topic string $SYS/Broker/+/warning/expiry/Subscription/#
2009-01-28 11:46:24.281:        None found.
2009-01-28 11:46:24.281: Migrating ACLs for topic string $SYS/Broker/+/Subscription/#
2009-01-28 11:46:24.281:        None found.
2009-01-28 11:46:24.281: Migrating ACLs for topic string $SYS/Broker/+/Status
2009-01-28 11:46:24.281:        None found.
2009-01-28 11:46:24.281: Migrating ACLs for topic string $SYS/Broker/+/Status/ExecutionGroup/#
2009-01-28 11:46:24.281:        None found.
2009-01-28 11:46:24.281: Migrating ACLs for topic string $SYS/STREAM/TEST_STREAM/RFH1/EU/France/Kersaux
2009-01-28 11:46:24.281: ... finished parsing topics
2009-01-28 11:46:24.281: Starting to parse retained publications ...
2009-01-28 11:46:24.296: Migrating retained publications for topic string $SYS/Broker/TEST_BROKER/Status
2009-01-28 11:46:24.296: Migrating retained publication for default subscription point.
2009-01-28 11:46:24.906: ... finished parsing retained publicatons
2009-01-28 11:46:24.968:
All Pub/Sub data has been retrieved from the broker.
2009-01-28 11:46:24.968: Applying changes to queue manager Pub/Sub state.
2009-01-28 11:46:24.972: Created topic object: MIGMBBRK.TOPIC.00004
2009-01-28 11:46:24.972: Created topic object: MIGMBBRK.TOPIC.00003
2009-01-28 11:46:24.972: Created topic object: MIGMBBRK.TOPIC.00002
2009-01-28 11:46:24.972: Created topic object: MIGMBBRK.TOPIC.00001
2009-01-28 11:46:24.977: Defining subscription [1]
2009-01-28 11:46:24.977: Defining subscription [2]
2009-01-28 11:46:24.977: Defining subscription [3]
2009-01-28 11:46:24.977: Defining subscription [4]
2009-01-28 11:46:24.993: Defining subscription [5]
2009-01-28 11:46:24.993: Defining subscription [6]
2009-01-28 11:46:24.993: Defining subscription [7]
2009-01-28 11:46:24.993: Defining subscription [8]
2009-01-28 11:46:29.996: Migration completion message written.
```

PROPCTL channel options

Use **PROPCTL** channel attribute to control which message properties are included in a message that is sent from a Version 7.5 queue manager to a queue manager from an earlier version of WebSphere MQ.

Table 100. Channel message property attribute settings

PROPCTL	**Description**
ALL	Use this option if applications connected to the Version 6.0 queue manager are able to process any properties placed in a message by a Version 7.5 application. All properties are sent to the Version 6.0 queue manager, in addition to any name/value pairs placed in the MQRFH2. You must consider two application design issues: 1. An application connected to the Version 6.0 queue manager must be able to process messages containing MQRFH2 headers generated on a Version 7.5 queue manager. 2. The application connected to the Version 6.0 queue manager must process new message properties that are flagged with MQPD_SUPPORT_REQUIRED correctly. With the ALL channel option set, JMS applications can interoperate between WebSphere MQ Version 6.0 and Version 7.5 using the channel. New Version 7.5 applications using message properties can interoperate with Version 6.0 applications, depending on how the Version 6.0 application handles MQRFH2 headers.

Table 100. Channel message property attribute settings (continued)

PROPCTL	**Description**
COMPAT	Use this option to send message properties to applications connected to a Version 6.0 queue manager in some cases, but not all. Message properties are only sent if two conditions are met: 1. No property must be marked as requiring message property processing. 2. At least one of the message properties must be in a “reserved” folder; see Note. With the COMPAT channel option set, JMS applications can interoperate between WebSphere MQ Version 6.0 and Version 7.5 using the channel. The channel is not available to every application using message properties, only to those applications that use the reserved folders. The rules concerning whether the message or the property is sent are: 1. If the message has properties, but none of the properties are associated with a “reserved” folder, then no message properties are sent. 2. If any message property has been created in a “reserved” property folder, all message properties associated with the message are sent. However: a. If any of the message properties are marked as support being required, MQPD_SUPPORT_REQUIRED or MQPD_SUPPORT_REQUIRED_IF_LOCAL, the whole message is rejected. It is returned, discarded, or sent to the dead letter queue according to the value of its report options. b. If no message properties are marked as support being required, an individual property might not be sent. If any of the message property descriptor fields are set to non-default values the individual property is not sent. The message is still sent. An example of a non-default property descriptor field value is MQPD_USER_CONTEXT. **Note:** The “reserved” folders names start with mcd., jms., usr., or mqext.. These folders are created for applications that use the JMS interface. In Version 7.5 any name/value pairs that are placed in these folders are treated as message properties. Message properties are sent in an MQRFH2 header, in addition to any name/value pairs placed in an MQRFH2 header. Any name/value pairs placed in an MQRFH2 header are sent as long as the message is not rejected.
NONE	Use this option to prevent any message properties being sent to applications connected to a Version 6.0 queue manager. An MQRFH2 that contains name/value pairs and message properties is still sent, but only with the name/value pairs. With the NONE channel option set, a JMS message is sent as a JMSTextMessage or a JMSBytesMessage without any JMS message properties. If it is possible for a Version 6.0 application to ignore all properties set in a Version 7.5 application, it can interoperate with it.

PROPCTL queue options

Use the **PROPCTL** queue attribute to control how message properties are returned to an application that calls MQGET without setting any MQGMO message property options.

Table 101. Queue message property attribute settings

PROPCTL	Description
ALL	Use this option so that different applications reading a message from the same queue can process the message in different ways. • An application, migrated unchanged from Version 6.0, can continue to read the MQRFH2 directly. Properties are directly accessible in the MQRFH2 header. You must modify the application to handle any new properties, and new property attributes. It is possible that the application might be affected by changes in the layout and number of MQRFH2 headers. Some folder attributes might be removed, or that WebSphere MQ reports an error in the layout of the MQRFH2 header that it ignored in Version 6.0. • A new or changed application can use the message property MQI to query message properties, and read name/value pairs in MQRFH2 header directly. All the properties in the message are returned to the application. • If the application calls MQCRTMH to create a message handle, it must query the message properties using MQINQMP. Name/value pairs that are not message properties remain in the MQRFH2, which is stripped of any message properties. • If the application does not create a message handle, all the message properties and name/value pairs remain in the MQRFH2. ALL only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.

Table 101. Queue message property attribute settings (continued)

PROPCTL	**Description**
COMPAT	COMPAT is the default option. If GMO_PROPERTIES_* is not set, as in an unmodified Version 6.0 application, COMPAT is assumed. By defaulting to the COMPAT option, a Version 6.0 application that did not explicitly create an MQRFH2, works without change on Version 7.5. Use this option if you have written an Version 6.0 MQI application to read JMS messages. • The JMS properties, which are stored in an MQRFH2 header, are returned to the application in an MQRFH2 header in folders with names starting with mcd., jms., usr., or mqext.. • If the message has JMS folders, and if a Version 7.5 application adds new property folders to the message, these properties are also returned in the MQRFH2. Consequently, you must modify the application to handle any new properties, and new property attributes. It is possible that an unmodified application might be affected by changes in the layout and number of MQRFH2 headers. It might find some folder attributes are removed, or that WebSphere MQ finds errors in the layout of the MQRFH2 header that it ignored in Version 6.0. **Note:** In this scenario, the behavior of the application is the same whether it is connected to a Version 6.0 or Version 7.5 queue manager. If the channel **PROPCTL** attribute is set to COMPAT or ALL any new message properties are sent in the message to the Version 6.0 queue manager. • If the message is not a JMS message, but contains other properties, those properties are not returned to the application in an MQRFH2 header.(The existence of specific property folders created by the WebSphere MQ classes for JMS indicates a JMS message. The property folders are mcd., jms., usr., or mqext.) • The option also enables Version 6.0 applications that explicitly create an MQRFH2 to work correctly, in many cases. For example, An MQI program that creates an MQRFH2 containing JMS message properties continues to work correctly. If a message is created without JMS message properties, but with some other MQRFH2 folders, the folders are returned to the application. Only if the folders are message property folders are those specific folders are removed from the MQRFH2. Message property folders are identified by having the new folder attribute content='properties', or are folders with names listed in Defined property folder name or Ungrouped property folder name. • If the application calls MQCRTMH to create a message handle, it must query the message properties using MQINQMP. Message properties are removed from the MQRFH2 headers. Name/value pairs that are not message properties remain in the MQRFH2. • If the application calls MQCRTMH to create a message handle, it can query all message properties, regardless of whether the message has JMS folders. • If the application does not create a message handle, all the message properties and name/value pairs remain in the MQRFH2. If a message contains new user property folders, you can infer that the message was created by a new or changed Version 7.5 application. If the receiving application is to process these new properties directly in an MQRFH2, you must modify the application to use the ALL option. With the default COMPAT option set, an unmodified application continues to process the rest of the MQRFH2, without the Version 7.5 properties. The intent of the PROPCTL interface is to support old applications reading MQRFH2 folders, and new and changed applications using the message property interface. Aim for new applications to use the message property interface for all user message properties, and to avoid reading and writing MQRFH2 headers directly. COMPAT only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.

Table 101. Queue message property attribute settings (continued)

PROPCTL	Description
FORCE	The FORCE option places all messages properties into MQRFH2 headers. All message properties and name/value pairs in the MQRFH2 headers remain in the message. Message properties are not removed from the MQRFH2, and made available through a message handle. The effect of choosing the FORCE option is to enable a newly migrated application to read message properties from MQRFH2 headers. Suppose you have modified an application to process Version 7.5 message properties, but have also retained its ability to work directly with MQRFH2 headers, as before. You can decide when to switch the application over to using message properties by initially setting the PROPCTL queue attribute to FORCE. Set the **PROPCTL** queue attribute to another value when you are ready to start using message properties. If the new function in the application does not behave as you expected, set the **PROPCTL** option back to FORCE. FORCE only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.
NONE	Use this option so that an existing application can process a message, ignoring all message properties, and a new or changed application can query message properties. • If the application calls MQCRTMH to create a message handle, it must query the message properties using MQINQMP. Name/value pairs that are not message properties remain in the MQRFH2, which is stripped of any message properties. • If the application does not create a message handle, all the message properties are removed from the MQRFH2. Name/value pairs in the MQRFH2 headers remain in the message. NONE only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.
V6COMPAT	Use this option to receive an MQRFH2 in the same format as it was sent. If the sending application, or the queue manager, creates additional message properties, these are returned in the message handle. This option has to be set on both the sending and receiving queues, and any intervening transmission queues. It overrides any other PROPCTL options set on queue definitions in the queue name resolution path. Use this option only in exceptional circumstances. For example, if you are migrating applications from Version 6.0 to Version 7.5 the option is valuable because it preserves Version 6.0 behavior. The option is likely to have an impact on message throughput. It is also more difficult to administer; you need to ensure the option is set on the sender, receiver, and intervening transmission queues. V6COMPAT only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.

Related information:

PROPCTL

MQGMO message property option settings

Use MQGMO message property options to control how message properties are returned to an application.

Table 102. MQGMO message property option settings

MQGMO Option	**Description**
MQGMO_PROPERTIES_AS_Q_DEF	Version 6.0 and Version 7.5 applications that read from the same queue, and do not set GMO_PROPERTIES_*, receive the message properties differently. Version 6.0 applications, and Version 7.5 applications that do not create a message handle, are controlled by the queue **PROPCTL** attribute. A Version 7.5 application can choose to receive message properties in the MQRFH2, or create a message handle and query the message properties. If the application creates a message handle, properties are removed from the MQRFH2. • An unmodified Version 6.0 application does not set GMO_PROPERTIES_*. Any message properties it receives are in the MQRFH2 headers. • A new or changed Version 7.5 application that does not set GMO_PROPERTIES_* or sets it to MQGMO_PROPERTIES_AS_Q_DEF can choose to query message properties. It must set MQCRTMH to create a message handle and query message properties using the MQINQMP MQI call. • If a new or changed application does not create a message handle, it behaves like a Version 6.0 application. It must read any message properties it receives directly from the MQRFH2 headers. • If the queue attribute **PROPCTL** is set to FORCE, no properties are returned in the message handle. All properties are returned in MQRFH2 headers. • If the queue attribute **PROPCTL** is set to NONE, or COMPAT, a Version 7.5 application that creates a message handle, receives all message properties. • If the queue attribute **PROPCTL** is set to V6COMPAT, and also set to V6COMPAT on all the queues the message was placed on between the sender and the receiver, properties set by MQSETMP are returned in the message handle, and properties and name/value pairs created in an MQRFH2 are returned in the MQRFH2. The format of an MQRFH2 sent in Version 7.5 is the same as Version 6.0, when sent by the same application.
MQGMO_PROPERTIES_IN_HANDLE	Force an application to use message properties. Use this option to detect if a modified application fails to create message handle. The application might be trying to read message properties directly from an MQRFH2, rather than call MQINQMP.
MQGMO_NO_PROPERTIES	Version 6.0 and a Version 7.5 application behave the same way, even if the Version 7.5 application creates a message handle. • All properties are removed. An unchanged Version 6.0 application connected to a Version 7.5 queue manager might behave differently to when it was connected to a Version 6.0 queue manager. Queue manager generated properties, such as JMS properties, are removed. • Properties are removed even if a message handle is created. Name/value pairs in other MQRFH2 folders are available in the message data.
MQGMO_PROPERTIES_FORCE_MQRFH2	Version 6.0 and Version 7.5 applications behave the same way. Properties are returned in the MQRFH2 headers, even if a message handle is created. • MQINQMP returns no message properties, even if a message handle is created. MQRC_PROPERTY_NOT_AVAILABLE is returned if a property is inquired upon.

Table 102. MQGMO message property option settings (continued)

MQGMO Option	Description
MQGMO_PROPERTIES_COMPATIBILITY	An Version 6.0 application connected to a Version 7.5 queue manager behaves the same as when it is connected to a Version 6.0 queue manager. If the message is from a JMS client, the JMS properties are returned in the MQRFH2 headers. New or modified Version 7.5 applications, that create a message handle, behave differently. • All properties in any message property folders are returned if the message contains a mcd., jms., usr., or mqext. folder. • If the message contains property folders, but not a mcd., jms., usr., or mqext. folder, no message properties are returned in an MQRFH2. • If a new or modified Version 7.5 application creates a message handle, query message properties using the MQINQMP MQI call. All message properties are removed from the MQRFH2. • If a new or modified Version 7.5 application creates a message handle, all properties in the message can be queried. Even if the message does not contain a mcd., jms., usr., or mqext. folder, all message properties are queriable.

Related information:

PROPCTL

2471 (09A7) (RC2471): MQRC_PROPERTY_NOT_AVAILABLE

Configuring

Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

Before you configure IBM WebSphere MQ, read about the IBM WebSphere MQ concepts in IBM WebSphere MQ Technical overview IBM WebSphere MQ Technical overview. Read about how to plan your IBM WebSphere MQ environment in Planning.

There are a number of different methods that you can use to create, configure, and administer your queue managers and their related resources in IBM WebSphere MQ. These methods include command line interfaces, a graphical user interface, and an administration API. For more information about these interfaces, see Administering IBM WebSphere MQ.

For instructions on how to create, start, stop, and delete a queue manager, see “Creating and managing queue managers” on page 751.

For information about how to create the components required to connect your IBM WebSphere MQ installations and applications together, see “Connecting applications using distributed queuing” on page 759.

For instructions on how to connect your clients to a IBM WebSphere MQ server by using different methods, see “Configuring connections between the server and client” on page 832.

For instructions on how to configure a queue manager cluster, see “Configuring a queue manager cluster” on page 885.

You can change the behavior of IBM WebSphere MQ or a queue manager by changing configuration information. For more information, see “Changing IBM WebSphere MQ and queue manager configuration information” on page 1143. In general, you do not need to restart a queue manager for any configuration changes to take effect, except for when stated in this product documentation.

Related information:

WebSphere MQ technical overview

Administering local WebSphere MQ objects

Administering remote WebSphere MQ objects

Planning
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Configuring multiple installations

When using multiple installations on the same system, you must configure the installations and queue managers.

This information applies to UNIX, Linux, and Windows.

Use the information in the following links to configure your installations:

- “Changing the primary installation” on page 747
- “Associating a queue manager with an installation” on page 748
- “Connecting applications in a multiple installation environment” on page 738

Related information:

Choosing a primary installation
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Multiple installations
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

Connecting applications in a multiple installation environment

On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

The following concepts are used to explain the way applications connect to IBM WebSphere MQ:

Linking
When the application is compiled, the application is linked to the IBM WebSphere MQ libraries to get the function exports that are then loaded when the application runs.

Loading
When the application is run, the IBM WebSphere MQ libraries are located and loaded. The specific mechanism used to locate the libraries varies by operating system, and by how the application is built. For more information about how to locate and load libraries in a multiple installation environment, see "Loading IBM WebSphere MQ Version 7.1 libraries" on page 740.

Connecting
When the application connects to a running queue manager, for example, using a `MQCONN` or `MQCONNX` call, it connects using the loaded IBM WebSphere MQ libraries.

When a server application connects to a queue manager, the loaded libraries must come from the installation associated with the queue manager. With multiple installations on a system, this restriction introduces new challenges when choosing the mechanism that the operating system uses to locate the IBM WebSphere MQ libraries to load:

- When the **setmqm** command is used to change the installation associated with a queue manager, the libraries that need to be loaded change.
- When an application connects to multiple queue managers that are owned by different installations, multiple sets of libraries need to be loaded.

However, if IBM WebSphere MQ Version 7.1, or later, libraries, are located and loaded, IBM WebSphere MQ then loads and uses the appropriate libraries without you needing to take any further action. When the application connects to a queue manager, IBM WebSphere MQ loads libraries from the installation that the queue manager is associated with.

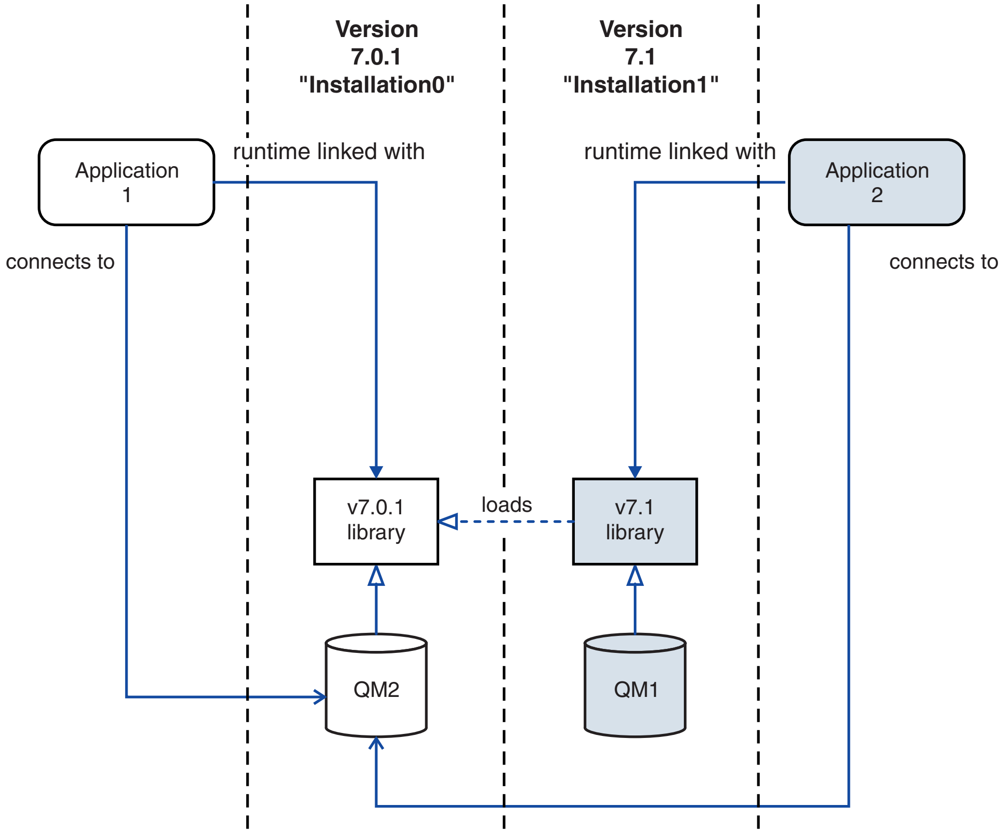


Figure 108. Connecting applications in a multiple installation environment

For example, Figure 108 shows a multiple installation environment with a version 7.0.1 installation (`Installation0`), and a version 7.1 installation (`Installation1`). Two applications are connected to these installations, but they load different library versions.

`Application 1` directly loads a version 7.0.1 library. When `application 1` connects to `QM2`, the version 7.0.1 libraries are used . If `application 1` attempts to connect to `QM1`, or if `QM2` is associated with `Installation1`, `application 1` fails with a 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE error. The application fails because the version 7.0.1 library is not capable of loading other library versions. That is, if version 7.0.1 libraries are directly loaded, you cannot use a queue manager associated with an installation at a later version of IBM WebSphere MQ.

`Application 2` directly loads a version 7.1 library. When `application 2` connects to `QM2`, the version 7.1 library then loads and uses the version 7.0.1 library. If `application 2` connects to `QM1`, or if `QM2` is associated with `Installation1`, the version 7.1 library is loaded, and the application works as expected.

Migration scenarios and connecting applications with multiple installations is considered in more detail in Multi-installation queue manager coexistence on UNIX, Linux, and Windows.

For more information about how to load IBM WebSphere MQ Version 7.1 libraries, see “Loading IBM WebSphere MQ Version 7.1 libraries” on page 740.

Support and restrictions

If any of the following version 7.1, or later, libraries, are located and loaded, IBM WebSphere MQ can automatically load and use the appropriate libraries:

- The C server libraries
- The C++ server libraries
- The XA server libraries
- The COBOL server libraries
- The COM+ server libraries
- .NET in unmanaged mode

IBM WebSphere MQ also automatically loads and uses the appropriate libraries for Java and JMS applications in bindings mode.

There are a number of restrictions for applications using multiple installations. For more information, see "Restrictions for applications using multiple installations" on page 743.

Related concepts:

"Associating a queue manager with an installation" on page 748
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

"Restrictions for applications using multiple installations" on page 743
There are restrictions when using CICS server libraries, fast path connections, message handles, and exits in a multiple installation environment.

"Loading IBM WebSphere MQ Version 7.1 libraries"
When deciding how to load IBM WebSphere MQ libraries, you need to consider a number of factors, including: your environment, whether you can change your existing applications, whether you want a primary installation, where IBM WebSphere MQ is installed, and whether the location of IBM WebSphere MQ is likely to change.

Related tasks:

"Changing the primary installation" on page 747
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Related information:

Choosing a primary installation
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Loading IBM WebSphere MQ Version 7.1 libraries

When deciding how to load IBM WebSphere MQ libraries, you need to consider a number of factors, including: your environment, whether you can change your existing applications, whether you want a primary installation, where IBM WebSphere MQ is installed, and whether the location of IBM WebSphere MQ is likely to change.

How IBM WebSphere MQ Version 7.1 libraries are located and loaded depends on your installation environment:

- On UNIX and Linux systems, if a copy of IBM WebSphere MQ Version 7.1 is installed in the default location, existing applications continue to work in the same way as previous versions. However, if the applications need symbolic links in `/usr/lib`, you must either select a version 7.1 installation to be the primary installation, or manually create the symbolic links.
- If IBM WebSphere MQ Version 7.1 is installed in a non-default location, which is the case if IBM WebSphere MQ Version 7.0.1 is also installed, you might need to change your existing applications so that the correct libraries are loaded.

How IBM WebSphere MQ Version 7.1, or later, libraries can be located and loaded also depends on how any existing applications are set up to load libraries. For more information about how libraries can be loaded, see “Operating system library loading mechanisms” on page 742.

Optimally, you should ensure that the queue manager is associated with the IBM WebSphere MQ library that is loaded by the operating system.

The methods for loading IBM WebSphere MQ libraries vary by platform, and each method has benefits and drawbacks.

Table 103. Benefits and drawbacks of the options for loading libraries

Platform	Option	Benefits	Drawbacks
UNIX and Linux systems	Set or change the embedded runtime search path (RPath) of the application. This option requires you to recompile and link the application. For more information about compiling and linking applications, see Building a WebSphere MQ application.	• Scope of the change is clear.	• You must be able to recompile and link the application. • If the location of IBM WebSphere MQ changes, you must change the RPath.
UNIX and Linux systems	Set the *LD_LIBRARY_PATH* environment variable (*LIBPATH* on AIX), using setmqenv, or crtmqenv, with the **-k** or **-l** option.	• No changes to existing applications required. • Overrides embedded RPaths in an application. • Easy to change the variable if the location of IBM WebSphere MQ changes.	• `setuid` and `setgid` applications, or applications built in other ways, might ignore *LD_LIBRARY_PATH* for security reasons. • Environment specific, so must be set in each environment where the application is run. • Possible impact on other applications that rely on *LD_LIBRARY_PATH*. • HP-UX: Options used when the application was compiled might disable the use of *LD_LIBRARY_PATH*. For more information, see Runtime linking considerations for HP-UX. • Linux: The compiler used to build the application might disable the use of *LD_LIBRARY_PATH*. For more information, see Runtime linking considerations for Linux.
Windows systems	Set the PATH variable using setmqenv, or crtmqenv.	• No changes required for existing applications. • Easy to change the variable if the location of IBM WebSphere MQ changes.	• Environment specific, so must be set in each environment where the application is run. • Possible impact on other applications.

Table 103. Benefits and drawbacks of the options for loading libraries (continued)

Platform	Option	Benefits	Drawbacks
UNIX, Linux, and Windows systems	Set the primary installation to a version 7.1, or later, installation. See "Changing the primary installation" on page 747. For more information about the primary installation, see Choosing a primary installation.	• No changes required for existing applications. • Easy to change the primary installation if the location of IBM WebSphere MQ changes. • Gives similar behavior to previous versions of IBM WebSphere MQ.	• When WebSphere MQ version 7.0.1 is installed, you cannot set the primary installation to version 7.1, or later. • UNIX and Linux: Does not work if `/usr/lib` is not in the default search path.

Library loading considerations for HP-UX

The sample compilation commands in the product documentation for previous versions of IBM WebSphere MQ included the `-Wl, +noenvvar` link option for 64-bit applications. This option disables the use of *LD_LIBRARY_PATH* to load shared libraries. If you want your applications to load IBM WebSphere MQ libraries from a location other than the location specified in the RPath, you must update your applications. You can update the applications by recompiling and linking without the `-Wl, +noenvvar` link option, or by using the **`chatr`** command.

To find out how your applications currently load libraries, see "Operating system library loading mechanisms."

Library loading considerations for Linux

Applications compiled using some versions of gcc, for example, version 3.2.x, can have an embedded RPath that cannot be overridden using the *LD_LIBRARY_PATH* environment variable. You can determine if an application is affected by using the `readelf -d` *`applicationName`* command. The RPath cannot be overridden if the RPATH symbol is present and the RUNPATH symbol is not present.

Library loading considerations for Solaris

The sample compilation commands in the product documentation for previous versions of IBM WebSphere MQ included the `-lmqmcs -lmqmzse` link options. The appropriate versions of these libraries are now loaded automatically by IBM WebSphere MQ. If IBM WebSphere MQ is installed in a non-default location, or if there are multiple installations on the system, you must update your applications. You can update the applications by recompiling and linking without the `-lmqmcs -lmqmzse` link options.

Operating system library loading mechanisms

On Windows systems, several directories are searched to find the libraries:

- The directory the application is loaded from.
- The current directory.
- The directories in the *PATH* environment variable, both the global *PATH* variable and the *PATH* variable of the current user.

On UNIX and Linux systems, there are a number of methods that might have been used to locate the libraries to load:

- Using the *LD_LIBRARY_PATH* environment variable (also *LIBPATH* on AIX, and *SHLIB_PATH* on HP-UX). If this variable is set, it defines a set of directories that are searched for the required

WebSphere MQ libraries. If any libraries are found in these directories, they are used in preference of any libraries that might be found using the other methods.

- Using an embedded search path (RPath). The application might contain a set of directories to search for the IBM WebSphere MQ libraries. If the *LD_LIBRARY_PATH* is not set, or if the required libraries were not found using the variable, the RPath is searched for the libraries. If your existing applications use an RPath, but you cannot recompile and link the application, you must either install IBM WebSphere MQ Version 7.1 in the default location, or use another method to find the libraries.
- Using the default library path. If the WebSphere MQ libraries are not found after searching the *LD_LIBRARY_PATH* variable and RPath locations, the default library path is searched. Usually, this path contains /usr/lib or /usr/lib64. If the libraries are not found after searching the default library path, the application fails to start because of missing dependencies.

You can use operating system mechanisms to find out if your applications have an embedded search path. For example:

- AIX: **dump**
- HP-UX: **chatr**
- Linux: **readelf**
- Solaris: **elfdump**

Related concepts:

"Associating a queue manager with an installation" on page 748
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

"Restrictions for applications using multiple installations"
There are restrictions when using CICS server libraries, fast path connections, message handles, and exits in a multiple installation environment.

"Connecting applications in a multiple installation environment" on page 738
On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

Related tasks:

"Changing the primary installation" on page 747
You can use the **setmqinst** command to set or unset an installation as the primary installation.

Related information:

Choosing a primary installation
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Restrictions for applications using multiple installations

There are restrictions when using CICS server libraries, fast path connections, message handles, and exits in a multiple installation environment.

CICS server libraries

If you are using the CICS server libraries, IBM WebSphere MQ does not automatically select the correct library level for you. You must compile and link your applications with the appropriate library level for the queue manager to which the application connects. For more information, see Building libraries for use with TXSeries® for Multiplatforms version 5.

Message handles

Message handles that use the special value of MQHC_UNASSOCIATED_HCONN are limited to use with the first installation loaded in a process. If the message handle cannot be used by a particular installation, reason code MQRC_HMSG_NOT_AVAILABLE is returned.

This restriction affects message properties. You cannot use message handles to get message properties from a queue manager on one installation and put them to a queue manager on a different installation. For more information about message handles, see MQCRTMH - Create message handle.

Exits

In a multiple installation environment, existing exits must be updated for use with IBM WebSphere MQ Version 7.1, or later, installations. Data conversion exits generated using the **crtmqcvx** command must be regenerated using the updated command.

All exits must be written using the MQIEP structure, cannot use an embedded RPATH to locate the IBM WebSphere MQ libraries, and cannot link to the IBM WebSphere MQ libraries. For more information, see Writing and compiling exits and installable services.

Fast path

On a server with multiple installations, applications using a fast path connection to IBM WebSphere MQ Version 7.1 or later must follow these rules:

1. The queue manager must be associated with the same installation as the one from which the application loaded the IBM WebSphere MQ run time libraries. The application must not use a fast path connection to a queue manager associated with a different installation. An attempt to make the connection results in an error, and reason code MQRC_INSTALLATION_MISMATCH.
2. Connecting non-fast path to a queue manager associated with the same installation as the one from which the application has loaded the IBM WebSphere MQ run time libraries prevents the application connecting fast path, unless either of these conditions are true:
   - The application makes its first connection to a queue manager associated with the same installation a fast path connection.
   - The environment variable, AMQ_SINGLE_INSTALLATION is set.
3. Connecting non-fast path to a queue manager associated with a Version 7.1 or later installation, has no effect on whether an application can connect fast path.
4. You cannot combine connecting to a queue manager associated with a Version 7.0.1 installation and connecting fast path to a queue manager associated with a Version 7.1, or later installation.

With AMQ_SINGLE_INSTALLATION set, you can make any connection to a queue manager a fast path connection. Otherwise almost the same restrictions apply:

- The installation must be the same one from which the IBM WebSphere MQ run time libraries were loaded.
- Every connection on the same process must be to the same installation. If you attempt to connect to a queue manager associated with a different installation, the connection fails with reason code MQRC_INSTALLATION_MISMATCH. Note that with AMQ_SINGLE_INSTALLATION set, this restriction applies to all connections, not only fast path connections.
- Only connect one queue manager with fast path connections.

Related information:

MQCONNX - Connect queue manager (extended)

MQIEP structure

2583 (0A17) (RC2583): MQRC_INSTALLATION_MISMATCH

2587 (0A1B) (RC2587): MQRC_HMSG_NOT_AVAILABLE

2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

Connecting .NET applications in a multiple installation environment

By default, applications use the .NET assemblies from the primary installation. If there is no primary installation, or you do not want to use the primary installation assemblies, you must update the application configuration file, or the *DEVPATH* environment variable.

If there is a primary installation on the system, the .NET assemblies and policy files of that installation are registered to the global assembly cache (GAC). The .NET assemblies for all other installations can be found in the installation path of each installation, but the assemblies are not registered to the GAC. Therefore, by default, applications run using the .NET assemblies from the primary installation. You must update the application configuration file if any of the following cases are true:

- You do not have a primary installation.
- You do not want the application to use the primary installation assemblies.
- The primary installation is a lower version of IBM WebSphere MQ than the version that the application was compiled with.

For information about how to update the application configuration file, see “Connecting .NET applications using the application configuration file.”

You must update the *DEVPATH* environment variable if the following case is true:

- You want your application to use the assemblies from a non-primary installation, but the primary installation is at the same version as the non-primary installation.

For more information about how to update the *DEVPATH* variable, see “Connecting .NET applications using DEVPATH” on page 746.

Connecting .NET applications using the application configuration file

Within the application configuration file, you must set various tags to redirect applications to use assemblies that are not from the primary installation.

The following table shows the specific changes that need to be made to the application configuration file to allow .NET applications connect using particular assemblies:

Table 104. Configuring applications to use particular assemblies

	Applications compiled with a lower version of IBM WebSphere MQ	**Applications compiled with a higher version of IBM WebSphere MQ**
To run an application with a higher version IBM WebSphere MQ primary installation. (higher version assemblies in GAC):	No changes necessary	No changes necessary
To run an application with a lower version IBM WebSphere MQ primary installation. (lower version assemblies in GAC):	No changes necessary	In the application configuration file: • Use the *<bindingRedirect>* tag to indicate the use of the lower version of the assemblies that are in the GAC

Table 104. Configuring applications to use particular assemblies (continued)

	Applications compiled with a lower version of IBM WebSphere MQ	**Applications compiled with a higher version of IBM WebSphere MQ**
To run an application with a higher version of IBM WebSphere MQ non-primary installation. (higher version assemblies in installation folder):	In the application configuration file: • Use the *<codebase>* tag to point to the location of the higher version assemblies • Use the *<bindingRedirect>* tag to indicate the use of the higher version assemblies	In the application configuration file: • Use the *<codebase>* tag to point to the location of the higher version assemblies
To run an application with a lower version of IBM WebSphere MQ non-primary installation. (lower version assemblies in installation folder):	In the application configuration file: • Use the *<codebase>* tag to point to the location of the lower version assemblies • Include the tag *<publisherpolicy Apply=no>*	In the application configuration file: • Use the *<codebase>* tag to point to the location of the lower version assemblies • Use the *<bindingRedirect>* tag to indicate the use of the lower version assemblies • Include the tag *<publisherpolicy Apply=no>*

A sample application configuration file `NonPrimaryRedirect.config` is shipped in the folder *MQ_INSTALLATION_PATH*`\tools\dotnet\samples\base`. This file can be modified with the IBM WebSphere MQ installation path of any non-primary installation. The file can also be directly included in other configuration files using the *<linkedConfiguration>* tag. Samples are provided for `nmqsget.exe.config` and `nmqsput.exe.config`. Both samples use the *<linkedConfiguration>* tag and include the `NonPrimaryRedirect.config` file.

Connecting .NET applications using DEVPATH

You can find the assemblies using the *DEVPATH* environment variable. The assemblies specified by the *DEVPATH* variable are used in preference to any assemblies in the GAC. See the appropriate Microsoft documentation on *DEVPATH* for more information about when to use this variable.

To find the assemblies using the *DEVPATH* environment variable, you must set the *DEVPATH* variable to the folder that contains the assemblies you want to use. Then, you must then update the application configuration file and add the following runtime configuration information:

```
<configuration>
   <runtime>
      <developmentMode developerInstallation="true" />
   </runtime>
</configuration>
```

Related concepts:

"Connecting applications in a multiple installation environment" on page 738
On UNIX, Linux, and Windows systems, if IBM WebSphere MQ Version 7.1, or later, libraries are loaded, IBM WebSphere MQ automatically uses the appropriate libraries without you needing to take any further action. IBM WebSphere MQ uses libraries from the installation associated with the queue manager that the application connects to.

Related information:

Choosing a primary installation
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

Using .NET

Multiple installations
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

Changing the primary installation

You can use the **setmqinst** command to set or unset an installation as the primary installation.

About this task

This task applies to UNIX, Linux, and Windows.

The primary installation is the installation to which required system-wide locations refer. For more information about the primary installation, and considerations for choosing your primary installation, see Choosing a primary installation.

If an installation of IBM WebSphere MQ Version 7.1 or later is coexisting with an installation of IBM WebSphere MQ Version 7.0.1, the IBM WebSphere MQ Version 7.0.1 installation must be the primary. It is flagged as primary when the IBM WebSphere MQ Version 7.1 or later version is installed, and the IBM WebSphere MQ Version 7.1 or later installation cannot be made primary.

During the installation process on Windows, you can specify that the installation is to be the primary installation. On UNIX and Linux systems, you must issue a **setmqinst** command after installation to set the installation as the primary installation.

"Set the primary installation."

"Unset the primary installation" on page 748.

Set the primary installation

Procedure

To set an installation as the primary installation:

1. Check if an installation is already the primary installation by entering the following command:
   ```
   MQ_INSTALLATION_PATH/bin/dspmqinst
   ```
   where *MQ_INSTALLATION_PATH* is the installation path of a IBM WebSphere MQ Version 7.1 or later installation.
2. If an existing IBM WebSphere MQ Version 7.1 or later installation is set as the primary installation, unset it by following the instructions in "Unset the primary installation" on page 748. If IBM WebSphere MQ Version 7.0.1 is installed on the system, the primary installation cannot be changed.
3. As root on UNIX and Linux systems, or a member of the Administrators group on Windows systems, enter one of the following commands:
   - To set the primary installation using the path of the installation you want to be the primary installation:
     ```
     MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
     ```
   - To set the primary installation using the name of the installation you want to be the primary installation:
     ```
     MQ_INSTALLATION_PATH/bin/setmqinst -i -n installationName
     ```
4. On Windows systems, restart the system.

Unset the primary installation

Procedure

To unset an installation as the primary installation:

1. Check which installation is the primary installation by entering the following command:

   ```
   MQ_INSTALLATION_PATH/bin/dspmqinst
   ```

   where *MQ_INSTALLATION_PATH* is the installation path of a IBM WebSphere MQ Version 7.1 or later installation.

   If IBM WebSphere MQ Version 7.0.1 is the primary installation, you cannot unset the primary installation.
2. As root on UNIX and Linux systems, or a member of the Administrators group on Windows systems, enter one of the following commands:
   - To unset the primary installation using the path of the installation you no longer want to be the primary installation:

     ```
     MQ_INSTALLATION_PATH/bin/setmqinst -x -p MQ_INSTALLATION_PATH
     ```
   - To unset the primary installation using the name of the installation you no longer want to be the primary installation:

     ```
     MQ_INSTALLATION_PATH/bin/setmqinst -x -n installationName
     ```

Related information:

Features that can be used only with the primary installation on Windows
Some Windows operating-system features can be used only with the primary installation. This restriction is due to the central registration of interface libraries, which might conflict as a result of multiple versions of IBM WebSphere MQ being installed.

External library and control command links to primary installation on UNIX and Linux
On UNIX and Linux platforms the primary installation is the one to which links from the `/usr` file system are made. However, only a subset of those links created with previous releases are now made.

Uninstalling, upgrading, and maintaining the primary installation
On all platforms, if you uninstall the primary installation, it ceases to be the primary installation. You must run the **setmqinst** command to select a new primary installation. On Windows, if you update the primary installation, it continues to be the primary installation. If you apply a fix pack to the primary installation, it continues to be the primary installation.

Choosing an installation name
Each installation of IBM WebSphere MQ on UNIX, Linux, and Windows, has a unique identifier known as an installation name. The installation name is used to associate things such as queue managers and configuration files with an installation.

setmqinst

Associating a queue manager with an installation

When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

You can use the **setmqm** command in the following ways:

- Moving individual queue managers between equivalent versions of WebSphere MQ. For example, moving a queue manager from a test to a production system.
- Migrating individual queue managers from an older version of WebSphere MQ to a newer version of WebSphere MQ. Migrating queue managers between versions has various implications that you must be aware of. For more information about migrating, see Migrating and upgrading WebSphere MQ.

To associate a queue manager with an installation:

1. Stop the queue manager using the **endmqm** command from the installation that is currently associated with the queue manager.
2. Associate the queue manager with another installation using the **setmqm** command from that installation.

   For example, to set queue manager QMB to be associated with an installation with the name Installation2, enter the following command from Installation2:

   ```
   MQ_INSTALLATION_PATH/bin/setmqm -m QMB -n Installation2
   ```

   where *MQ_INSTALLATION_PATH* is the path where Installation2 is installed.
3. Start the queue manager using the **strmqm** command from the installation that is now associated with the queue manager.

   This command performs any necessary queue manager migration and results in the queue manager being ready to use.

The installation that a queue manager is associated with limits that queue manager so that it can be administered only by commands from that installation. There are three key exceptions:

- **setmqm** changes the installation associated with the queue manager. This command must be issued from the installation that you want to associate with the queue manager, not the installation that the queue manager is currently associated with. The installation name specified by the **setmqm** command has to match the installation from which the command is issued.
- **strmqm** usually has to be issued from the installation that is associated with the queue manager. However, when a V7.0.1 or earlier queue manager is started on a V7.1 or later installation for the first time, **strmqm** can be used. In this case, **strmqm** starts the queue manager and associates it with the installation from which the command is issued.
- **dspmq** displays information about all queue managers on a system, not just those queue managers associated with the same installation as the **dspmq** command. The `dspmq -o installation` command displays information about which queue managers are associated with which installations.

Queue manager association in HA environments

For HA environments, the **addmqinf** command automatically associates the queue manager with the installation from which the **addmqinf** command is issued. As long as the **strmqm** command is then issued from the same installation as the **addmqinf** command, no further setup is required. To start the queue manager using a different installation, you must first change the associated installation using the **setmqm** command.

Queue managers associated with deleted installations

If the installation that a queue manager is associated with has been deleted, or if the queue manager status information is unavailable, the **setmqm** command fails to associate the queue manager with another installation. In this situation, take the following actions:

1. Use the **dspmqinst** command to see the other installations on your system.
2. Manually modify the `InstallationName` field of the `QueueManager` stanza in mqs.ini to specify another installation.
3. Use the **dltmqm** command from that installation to delete the queue manager.

Related concepts:

"Finding installations of IBM WebSphere MQ on a system"
If you have multiple IBM WebSphere MQ installations on a system, you can check which versions are installed and where they are.

"The IBM WebSphere MQ configuration file, mqs.ini" on page 1145
The IBM WebSphere MQ configuration file, mqs.ini, contains information relevant to all the queue managers on the node. It is created automatically during installation.

Related information:

Choosing a primary installation
On systems that support multiple installations of IBM WebSphere MQ (UNIX, Linux, and Windows), the primary installation is the one to which IBM WebSphere MQ system-wide locations refer. Having a primary installation is optional, but convenient.

setmqm

strmqm

dspmq

dspmqinst

Finding installations of IBM WebSphere MQ on a system

If you have multiple IBM WebSphere MQ installations on a system, you can check which versions are installed and where they are.

You can use the following methods to find the IBM WebSphere MQ installations on your system:

- Use the **dspmqver** command. This command does not provide details of all installations on a system if it is issued from a Version 7.0.1 installation.
- Use the platform installation tools to query where IBM WebSphere MQ has been installed. Then use the **dspmqver** command from a Version 7.1 or later installation. The following commands are examples of commands you can use to query where IBM WebSphere MQ has been installed:
  - On AIX systems, you can use the **lslpp** command:
    ```
    lslpp -R ALL -l mqm.base.runtime
    ```
  - On HP-UX systems, you can use the **swlist** command:
    ```
    swlist -a location -a revision -l product MQSERIES
    ```
  - On Linux systems, you can use the **rpm** command:
    ```
    rpm -qa --qf "%{NAME}-%{VERSION}-%{RELEASE}\t%{INSTPREFIXES}\n" | grep MQSeriesRuntime
    ```
  - On Solaris systems, you can use the **pkginfo** and **pkgparam** commands:
    1. List the installed packages by entering the following command:
       ```
       pkginfo | grep -w mqm
       ```
    2. For each package listed, enter following command:
       ```
       pkgparam pkgname BASEDIR
       ```
  - On Windows systems, you can use the **wmic** command. This command might install the wmic client:
    ```
    wmic product where "(Name like '%MQ%') AND (not Name like '%bitSupport')" get Name, Version, InstallLocation
    ```
- On UNIX and Linux systems, issue the following command to find out where IBM WebSphere MQ has been installed:
  ```
  cat /etc/opt/mqm/mqinst.ini
  ```

  Then use the **dspmqver** command from a Version 7.1 or later installation.
- To display details of installations on the system, on 32-bit Windows, issue the following command:
  ```
  reg.exe query "HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere MQ\Installation" /s
  ```
- On 64-bit Windows, issue the following command:
  ```
  reg.exe query "HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\IBM\WebSphere MQ\Installation" /s
  ```

Note: the **reg.exe** command will only display information for Version 7.1 or later installations.

Related information:

dspmqver

dspmqinst

Multiple installations
On UNIX, Linux, and Windows, it is possible to have more than one copy of IBM WebSphere MQ on a system.

Creating and managing queue managers

Before you can use messages and queues, you must create and start at least one queue manager and its associated objects.

Creating a queue manager

A queue manager manages the resources associated with it, in particular the queues that it owns. It provides queuing services to applications for Message Queuing Interface (MQI) calls and commands to create, modify, display, and delete IBM WebSphere MQ objects.

To create a queue manager, use the IBM WebSphere MQ control command **crtmqm** (described in **crtmqm**). The **crtmqm** command automatically creates the required default objects and system objects (described in System default objects). Default objects form the basis of any object definitions that you make; system objects are required for queue manager operation. When you have created a queue manager and its objects, use the **strmqm** command to start the queue manager.

Note: IBM WebSphere MQ does not support machine names that contain spaces. If you install IBM WebSphere MQ on a computer with a machine name that contains spaces, you cannot create any queue managers.

On

Before you can create a queue manager, there are several points you must consider (especially in a production environment). Work through the following checklist:

The installation associated with the queue manager
The **crtmqm** command automatically associates a queue manager with the installation from which the **crtmqm** command was issued. For commands that operate on a queue manager, you must issue the command from the installation associated with the queue manager. You can change the associated installation of a queue manager using the setmqm command. Note the Windows installer does not add the user that performs the install to the mqm group, for more details, see Authority to administer WebSphere MQ on UNIX, Linux and Windows systems.

Naming conventions
Use uppercase names so that you can communicate with queue managers on all platforms. Remember that names are assigned exactly as you enter them. To avoid the inconvenience of lots of typing, do not use unnecessarily long names.

Specify a unique queue manager name
When you create a queue manager, ensure that no other queue manager has the same name *anywhere* in your network. Queue manager names are not checked when the queue manager is created, and names that are not unique prevent you from creating channels for distributed queuing.

One way of ensuring uniqueness is to prefix each queue manager name with its own unique node name. For example, if a node is called `ACCOUNTS`, you can name your queue manager `ACCOUNTS.SATURN.QUEUE.MANAGER`, where `SATURN` identifies a particular queue manager and

`QUEUE.MANAGER` is an extension you can give to all queue managers. Alternatively, you can omit this, but note that `ACCOUNTS.SATURN` and `ACCOUNTS.SATURN.QUEUE.MANAGER` are *different* queue manager names.

If you are using IBM WebSphere MQ for communication with other enterprises, you can also include your own enterprise name as a prefix. This is not done in the examples, because it makes them more difficult to follow.

Note: Queue manager names in control commands are case-sensitive. This means that you are allowed to create two queue managers with the names `jupiter.queue.manager` and `JUPITER.queue.manager`. However, it is better to avoid such complications.

Limit the number of queue managers

You can create as many queue managers as resources allow. However, because each queue manager requires its own resources, it is generally better to have one queue manager with 100 queues on a node than to have ten queue managers with ten queues each.

In production systems, many processors can be exploited with a single queue manager, but larger server machines might run more effectively with multiple queue managers.

Specify a default queue manager

Each node should have a default queue manager, though it is possible to configure IBM WebSphere MQ on a node without one. The default queue manager is the queue manager that applications connect to if they do not specify a queue manager name in an **MQCONN** call. It is also the queue manager that processes MQSC commands when you invoke the **runmqsc** command without specifying a queue manager name.

Specifying a queue manager as the default *replaces* any existing default queue manager specification for the node.

Changing the default queue manage can affect other users or applications. The change has no effect on currently-connected applications, because they can use the handle from their original connect call in any further MQI calls. This handle ensures that the calls are directed to the same queue manager. Any applications connecting *after* you have changed the default queue manager connect to the new default queue manager. This might be what you intend, but you should take this into account before you change the default.

Creating a default queue manager is described in “Creating a default queue manager” on page 754.

Specify a dead-letter queue

The dead-letter queue is a local queue where messages are put if they cannot be routed to their intended destination.

It is important to have a dead-letter queue on each queue manager in your network. If you do not define one, errors in application programs might cause channels to be closed, and replies to administration commands might not be received.

For example, if an application tries to put a message on a queue on another queue manager, but gives the wrong queue name, the channel is stopped and the message remains on the transmission queue. Other applications cannot then use this channel for their messages.

The channels are not affected if the queue managers have dead-letter queues. The undelivered message is put on the dead-letter queue at the receiving end, leaving the channel and its transmission queue available.

When you create a queue manager, use the -u flag to specify the name of the dead-letter queue. You can also use an MQSC command to alter the attributes of a queue manager that you have already defined to specify the dead-letter queue to be used. See Working with queue managers for an example of the MQSC command ALTER.

Specify a default transmission queue

A transmission queue is a local queue on which messages in transit to a remote queue manager are queued before transmission. The default transmission queue is the queue that is used when no transmission queue is explicitly defined. Each queue manager can be assigned a default transmission queue.

When you create a queue manager, use the -d flag to specify the name of the default transmission queue. This does not actually create the queue; you have to do this explicitly later on. See Working with local queues for more information.

Specify the logging parameters you require

You can specify logging parameters on the **crtmqm** command, including the type of logging, and the path and size of the log files.

In a development environment, the default logging parameters should be adequate. However, you can change the defaults if, for example:

- You have a low-end system configuration that cannot support large logs.
- You anticipate a large number of long messages being on your queues at the same time.
- You anticipate a lot of persistent messages passing through the queue manager.

Once you have set the logging parameters, some of them can only be changed by deleting the queue manager and recreating it with the same name but with different logging parameters.

For more information about logging parameters, see "Availability, recovery and restart" on page 1034.

For IBM WebSphere MQ for UNIX systems only

You can create the queue manager directory `/var/mqm/qmgrs/<qmgr>`, even on a separate local file system, before you use the **crtmqm** command. When you use **crtmqm**, if the `/var/mqm/qmgrs/<qmgr>` directory exists, is empty, and is owned by mqm, it is used for the queue manager data. If the directory is not owned by mqm, the creation fails with a First Failure Support Technology (FFST) message. If the directory is not empty, a new directory is created.

Related concepts:

"Configuring" on page 737
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

"Backing up configuration files after creating a queue manager" on page 756
IBM WebSphere MQ configuration information is stored in configuration files on Windows, UNIX and Linux systems.

"Starting a queue manager" on page 756
When you create a queue manager, you must start it to enable it to process commands or MQI calls.

"Stopping a queue manager" on page 757
There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

"Restarting a queue manager" on page 758
You can use the **`strmqm`** command to restart a queue manager, or, on IBM WebSphere MQ for Windows and IBM WebSphere MQ for Linux (x86 and x86-64 platforms) systems, restart a queue manager from IBM WebSphere MQ Explorer.

"Changing IBM WebSphere MQ and queue manager configuration information" on page 1143
Change the behavior of IBM WebSphere MQ or an individual queue manager to suit the needs of your installation.

Related tasks:

"Making an existing queue manager the default" on page 755
You can make an existing queue manager the default queue manager. The way you do this depends on

the platform you are using.

"Deleting a queue manager" on page 758
You can delete a queue manager using the **dltmqm** command or by using the WebSphere MQ Explorer.

Related information:

System and default objects

Creating a default queue manager

The default queue manager is the queue manager that applications connect to if they do not specify a queue manager name in an MQCONN call. It is also the queue manager that processes MQSC commands when you invoke the **runmqsc** command without specifying a queue manager name. To create a queue manager, you use the IBM WebSphere MQ control command **crtmqm**.

Before you begin

Before creating a default queue manager, read through the considerations described in "Creating and managing queue managers" on page 751.

UNIX When you use **crtmqm** to create a queue manager on UNIX and Linux, if the /var/mqm/qmgrs/<qmgr> directory already exists, is owned by mqm, and is empty, it is used for the queue manager data. If the directory is not owned by mqm, the creation of the queue manager fails with a First Failure Support Technology (FFST) message. If the directory is not empty, a new directory is created for the queue manager data.

This consideration applies even when the /var/mqm/qmgrs/<qmgr> directory already exists on a separate local file system.

About this task

When you create a queue manager by using the **crtmqm** command, the command automatically creates the required default objects and system objects. Default objects form the basis of any object definitions that you make and system objects are required for queue manager operation.

By including the relevant parameters in the command, you can also define, for example, the name of the default transmission queue to be used by the queue manager, and the name of the dead letter queue.

Windows On Windows, you can use the **sax** option of the **crtmqm** command to start multiple instances of the queue manager.

For more information about the **crtmqm** command and its syntax, see **crtmqm**.

Procedure

To create a default queue manager, use the **crtmqm** command with the **-q** flag. The following example of the **crtmqm** command creates a default queue manager called SATURN.QUEUE.MANAGER:

```
crtmqm -q -d MY.DEFAULT.XMIT.QUEUE -u SYSTEM.DEAD.LETTER.QUEUE SATURN.QUEUE.MANAGER
```

where:

-q Indicates that this queue manager is the default queue manager.

-d MY.DEFAULT.XMIT.QUEUE
Is the name of the default transmission queue to be used by this queue manager.

Note: IBM WebSphere MQ does not create a default transmission queue for you; you have to define it yourself.

-u SYSTEM.DEAD.LETTER.QUEUE
Is the name of the default dead-letter queue created by IBM WebSphere MQ on installation.

SATURN.QUEUE.MANAGER
Is the name of this queue manager. This must be the last parameter specified on the **crtmqm** command.

What to do next

When you have created a queue manager and its objects, use the **strmqm** command to start the queue manager.

Related concepts:

"Backing up configuration files after creating a queue manager" on page 756
IBM WebSphere MQ configuration information is stored in configuration files on Windows, UNIX and Linux systems.

Related information:

Working with queue managers

Working with local queues

System and default objects

Making an existing queue manager the default

You can make an existing queue manager the default queue manager. The way you do this depends on the platform you are using.

WebSphere MQ for Windows and WebSphere MQ for Linux (x86 and x86-64 platforms) systems

About this task

Use the following instructions to make an existing queue manager the default queue manager on WebSphere MQ for Windows and WebSphere MQ for Linux (x86 and x86-64 platforms) systems:

Procedure

1. Open the WebSphere MQ Explorer.
2. Right-click **IBM WebSphere MQ**, then select **Properties...**. The Properties for WebSphere MQ panel is displayed.
3. Type the name of the default queue manager into the Default queue manager name field.
4. Click OK.

UNIX and Linux systems

About this task

When you create a default queue manager, its name is inserted in the `Name` attribute of the `DefaultQueueManager` stanza in the WebSphere MQ configuration file (mqs.ini). The stanza and its contents are automatically created if they do not exist.

Procedure

- To make an existing queue manager the default, change the queue manager name on the `Name` attribute to the name of the new default queue manager. You can do this manually, using a text editor.
- If you do not have a default queue manager on the node, and you want to make an existing queue manager the default, create the *`DefaultQueueManager`* stanza with the required name yourself.

- If you accidentally make another queue manager the default and want to revert to the original default queue manager, edit the `DefaultQueueManager` stanza in mqs.ini, replacing the unwanted default queue manager with that of the one you want.

What to do next

See "Changing IBM WebSphere MQ and queue manager configuration information" on page 1143 for information about configuration files.

Backing up configuration files after creating a queue manager

IBM WebSphere MQ configuration information is stored in configuration files on Windows, UNIX and Linux systems.

On Windows and Linux (x86 and x86-64) systems use IBM WebSphere MQ Explorer to make changes to the configuration files.

On Windows systems you can also use the **amqmdain** command to make changes to the configuration files. See, amqmdain

There are two types of configuration file:

- When you install the product, the IBM WebSphere MQ configuration file (mqs.ini) is created. It contains a list of queue managers that is updated each time you create or delete a queue manager. There is one mqs.ini file per node.
- When you create a new queue manager, a new queue manager configuration file (qm.ini) is automatically created. This contains configuration parameters for the queue manager.

After creating a queue manager, back up your configuration files. Then, if you create another queue manager that causes you problems, you can reinstate the backups when you have removed the source of the problem. As a general rule, back up your configuration files each time you create a new queue manager.

For more information about configuration files, see "Changing IBM WebSphere MQ and queue manager configuration information" on page 1143.

Starting a queue manager

When you create a queue manager, you must start it to enable it to process commands or MQI calls.

To start a queue manager, use the **`strmqm`** command.

Note: You must use the **`strmqm`** command from the installation associated with the queue manager that you are working with. You can find out which installation a queue manager is associated with using the `dspmq -o installation` command.

For example, to start a queue manager `QMB` enter the following command:

```
strmqm QMB
```

On WebSphere MQ for Windows and WebSphere MQ for Linux (x86 and x86-64 platforms) systems, you can start a queue manager as follows:

1. Open the WebSphere MQ Explorer.
2. Select the queue manager from the Navigator View.
3. Click **Start**. The queue manager starts.

If the queue manager start-up takes more than a few seconds WebSphere MQ issues information messages intermittently detailing the start-up progress.

The **strmqm** command does not return control until the queue manager has started and is ready to accept connection requests.

Starting a queue manager automatically

In WebSphere MQ for Windows you can start a queue manager automatically when the system starts using the WebSphere MQ Explorer. For more information, see Administration using the IBM WebSphere MQ Explorer.

Stopping a queue manager

There are three ways to stop a queue manager: a quiesced shutdown, and immediate shutdown, and a preemptive shutdown.

Use the **endmqm** command to stop a queue manager.

Note: You must use the **endmqm** command from the installation associated with the queue manager that you are working with. You can find out which installation a queue manager is associated with using the `dspmq -o installation` command.

For example, to stop a queue manager called `QMB`, enter the following command:

```
endmqm QMB
```

On WebSphere MQ for Windows and WebSphere MQ for Linux (x86 and x86-64 platforms) systems, you can stop a queue manager as follows:

1. Open the WebSphere MQ Explorer.
2. Select the queue manager from the Navigator View.
3. Click **Stop...**. The End Queue Manager panel is displayed.
4. Select Controlled, or Immediate.
5. Click **OK**. The queue manager stops.

Quiesced shutdown

By default, the **endmqm** command performs a quiesced shutdown of the specified queue manager. This might take a while to complete. A quiesced shutdown waits until all connected applications have disconnected.

Use this type of shutdown to notify applications to stop. If you issue:

```
endmqm -c QMB
```

you are not told when all applications have stopped. (An `endmqm -c QMB` command is equivalent to an `endmqm QMB` command.)

However, if you issue:

```
endmqm -w QMB
```

the command waits until all applications have stopped and the queue manager has ended.

Immediate shutdown

For an immediate shutdown any current MQI calls are allowed to complete, but any new calls fail. This type of shutdown does not wait for applications to disconnect from the queue manager.

For an immediate shutdown, type:

```
endmqm -i QMB
```

Preemptive shutdown

Note: Do not use this method unless all other attempts to stop the queue manager using the **endmqm** command have failed. This method can have unpredictable consequences for connected applications.

If an immediate shutdown does not work, you must resort to a *preemptive* shutdown, specifying the `-p` flag. For example:

```
endmqm -p QMB
```

This stops the queue manager immediately. If this method still does not work, see Stopping a queue manager manually for an alternative solution.

For a detailed description of the **endmqm** command and its options, see endmqm.

If you have problems shutting down a queue manager

Problems in shutting down a queue manager are often caused by applications. For example, when applications:

- Do not check MQI return codes properly
- Do not request notification of a quiesce
- Terminate without disconnecting from the queue manager (by issuing an **MQDISC** call)

If a problem occurs when you stop the queue manager, you can break out of the **endmqm** command using Ctrl-C. You can then issue another **endmqm** command, but this time with a flag that specifies the type of shutdown that you require.

Restarting a queue manager

You can use the **strmqm** command to restart a queue manager, or, on IBM WebSphere MQ for Windows and IBM WebSphere MQ for Linux (x86 and x86-64 platforms) systems, restart a queue manager from IBM WebSphere MQ Explorer.

To restart a queue manager, type:

```
strmqm saturn.queue.manager
```

On IBM WebSphere MQ for Windows and IBM WebSphere MQ for Linux (x86 and x86-64 platforms) systems, you can restart a queue manager in the same way as starting it, as follows:

1. Open the IBM WebSphere MQ Explorer.
2. Select the queue manager from the Navigator View.
3. Click **Start**. The queue manager restarts.

If the queue manager restart takes more than a few seconds IBM WebSphere MQ issues information messages intermittently detailing the start-up progress.

Deleting a queue manager

You can delete a queue manager using the **dltmqm** command or by using the WebSphere MQ Explorer.

Before you begin

Stop the queue manager.

Procedure

Issue the following command: `dltmqm QMB`

Note: You must use the **dltmqm** command from the installation associated with the queue manager that you are working with. You can find out which installation a queue manager is associated with using the `dspmq -o installation` command.

Steps for deleting a queue manager

About this task

On WebSphere MQ for Windows and WebSphere MQ for Linux (x86 and x86-64 platforms) systems, you can delete a queue manager as follows:

Procedure

1. Open the WebSphere MQ Explorer.
2. In the Navigator view, select the queue manager.
3. If the queue manager is not stopped, stop it.
   a. Right-click the queue manager.
   b. Click **Stop**.
4. Right-click the queue manager.
5. Click **Delete**.

Results

The queue manager is deleted.

Attention:

- Deleting a queue manager is a drastic step, because you also delete all resources associated with the queue manager, including all queues and their messages and all object definitions. If you use the **dltmqm** command, there is no displayed prompt that allows you to change your mind; when you press the Enter key all the associated resources are lost.
- In WebSphere MQ for Windows, deleting a queue manager also removes the queue manager from the automatic startup list (described in "Starting a queue manager" on page 756). When the command has completed, a `WebSphere MQ queue manager ending` message is displayed; you are not told that the queue manager has been deleted.
- Deleting a cluster queue manager does not remove it from the cluster. See the note in the description of **dltmqm** for more information.

For a description of the **dltmqm** command and its options, see dltmqm. Ensure that only trusted administrators have the authority to use this command. (For information about security, see Setting up security on Windows, UNIX and Linux systems .)

Connecting applications using distributed queuing

This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

Before reading this section it is helpful to have an understanding of channels, queues, and the other concepts introduced in Concepts of intercommunication .

Use the information in the following links to connect your applications using distributed queuing:

- "How to send a message to another queue manager" on page 783
- "Triggering channels" on page 801
- "Safety of messages" on page 799
- "WebSphere MQ distributed-messaging techniques" on page 760

- “Introduction to distributed queue management” on page 780
- Windows UNIX Linux “Monitoring and controlling channels on Windows, UNIX and Linux platforms” on page 807

Related concepts:

“Configuring connections between the server and client” on page 832
To configure the communication links between WebSphere MQ MQI clients and servers, decide on your communication protocol, define the connections at both ends of the link, start a listener, and define channels.

“Changing IBM WebSphere MQ and queue manager configuration information” on page 1143
Change the behavior of IBM WebSphere MQ or an individual queue manager to suit the needs of your installation.

Related tasks:

“Configuring a queue manager cluster” on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

WebSphere MQ distributed-messaging techniques

The subtopics in this section describe techniques that are of use when planning channels. These subtopics describe techniques to help you plan how to connect your queue managers together, and manage the flow of messages between your applications.

For message channel planning examples, see:

- Message channel planning example for distributed platforms

Related concepts:

“Connecting applications using distributed queuing” on page 759
This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

Related information:

Channels

Introduction to message queuing

Concepts of intercommunication

Example configuration information

Message flow control

Message flow control is a task that involves the setting up and maintenance of message routes between queue managers. It is important for routes that multi-hop through many queue managers. This section describes how you use queues, alias queue definitions, and message channels on your system to achieve message flow control.

You control message flow using a number of techniques that were introduced in “Connecting applications using distributed queuing” on page 759. If your queue manager is in a cluster, message flow is controlled using different techniques, as described in “Message flow control.”

You can use the following objects to achieve message flow control:

- Transmission queues
- Message channels
- Remote queue definition
- Queue manager alias definition
- Reply-to queue alias definition

The queue manager and queue objects are described in Objects. Message channels are described in Distributed queuing components. The following techniques use these objects to create message flows in your system:

- Putting messages to remote queues
- Routing by way of particular transmission queues
- Receiving messages
- Passing messages through your system
- Separating message flows
- Switching a message flow to another destination
- Resolving the reply-to queue name to an alias name

Note

All the concepts described in this section are relevant for all nodes in a network, and include sending and receiving ends of message channels. For this reason, only one node is illustrated in most examples. The exception is where the example requires explicit cooperation by the administrator at the other end of a message channel.

Before proceeding to the individual techniques, it is useful to recap on the concepts of name resolution and the three ways of using remote queue definitions. See Concepts of intercommunication.

Related concepts:

"Queue names in transmission header"
Destination queue names travel with the message in the transmission header until the destination queue has been reached.

"How to create queue manager and reply-to aliases"
This topic explains the three ways that you can create a remote queue definition.

Queue names in transmission header:

Destination queue names travel with the message in the transmission header until the destination queue has been reached.

The queue name used by the application, the logical queue name, is resolved by the queue manager to the destination queue name. In other words, the physical queue name. This destination queue name travels with the message in a separate data area, the transmission header, until the destination queue has been reached. The transmission header is then stripped off.

You change the queue manager part of this queue name when you create parallel classes of service. Remember to return the queue manager name to the original name when the end of the class-of-service diversion has been reached.

How to create queue manager and reply-to aliases:

This topic explains the three ways that you can create a remote queue definition.

The remote queue definition object is used in three different ways. Table 105 on page 762 explains how to define each of the three ways:

- Using a remote queue definition to redefine a local queue name.

  The application provides only the queue name when opening a queue, and this queue name is the name of the remote queue definition.

  The remote queue definition contains the names of the target queue and queue manager. Optionally, the definition can contain the name of the transmission queue to be used. If no transmission queue name is provided, the queue manager uses the queue manager name, taken from the remote queue definition, for the transmission queue name. If a transmission queue of this name is not defined, but a default transmission queue is defined, the default transmission queue is used.

- Using a remote queue definition to redefine a queue manager name.

  The application, or channel program, provides a queue name together with the remote queue manager name when opening the queue.

  If you have provided a remote queue definition with the same name as the queue manager name, and you have left the queue name in the definition blank, then the queue manager substitutes the queue manager name in the open call with the queue manager name in the definition.

  In addition, the definition can contain the name of the transmission queue to be used. If no transmission queue name is provided, the queue manager takes the queue manager name, taken from the remote queue definition, for the transmission queue name. If a transmission queue of this name is not defined, but a default transmission queue is defined, the default transmission queue is used.

- Using a remote queue definition to redefine a reply-to queue name.

  Each time an application puts a message to a queue, it can provide the name of a reply-to queue for answer messages but with the queue manager name blank.

  If you provide a remote queue definition with the same name as the reply-to queue then the local queue manager replaces the reply-to queue name with the queue name from your definition.

  You can provide a queue manager name in the definition, but not a transmission queue name.

Table 105. Three ways of using the remote queue definition object

Usage	**Queue manager name**	**Queue name**	**Transmission queue name**
1. Remote queue definition (on OPEN call)			
Supplied in the call	blank or local QM	(*) required	-
Supplied in the definition	required	required	optional
2. Queue manager alias (on OPEN call)			
Supplied in the call	(*) required and not local QM	required	-
Supplied in the definition	required	blank	optional
3. Reply-to queue alias (on PUT call)			
Supplied in the call	blank	(*) required	-
Supplied in the definition	optional	optional	blank
Note: (*) means that this name is the name of the definition object			

For a formal description, see Queue name resolution.

Putting messages on remote queues

You can use remote queue definition objects to resolve a queue name to a transmission queue to an adjacent queue manager.

In a distributed-queuing environment, a transmission queue and channel are the focal point for all messages to a location whether the messages originate from applications in your local system, or arrive through channels from an adjacent system. Figure 109 on page 763 shows an application placing messages on a logical queue named 'QA_norm'. The name resolution uses the remote queue definition 'QA_norm' to select the transmission queue QMB. It then adds a transmission header to the messages stating 'QA_norm at QMB'.

Messages arriving from the adjacent system on 'Channel_back' have a transmission header with the physical queue name 'QA_norm at QMB', for example. These messages are placed unchanged on transmission queue QMB.

The channel moves the messages to an adjacent queue manager.

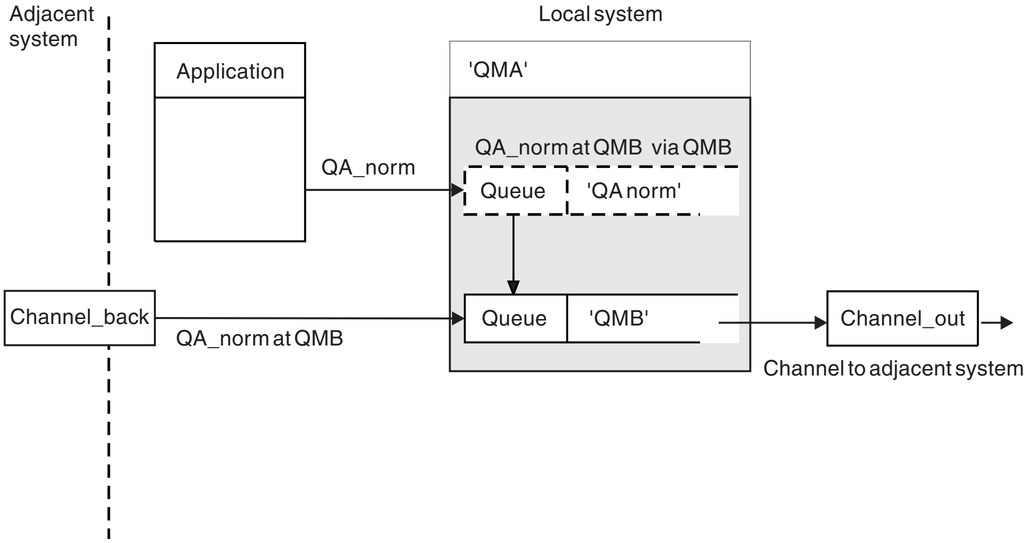


Figure 109. A remote queue definition is used to resolve a queue name to a transmission queue to an adjacent queue manager. Note: The dashed outline represents a remote queue definition. This queue is not a real queue, but a name alias that is controlled as though it were a real queue.

If you are the WebSphere MQ system administrator, you must:

- Define the message channel from the adjacent system
- Define the message channel to the adjacent system
- Create the transmission queue QMB
- Define the remote queue object 'QA_norm' to resolve the queue name used by applications to the destination queue name, destination queue manager name, and transmission queue name

In a clustering environment, you only need to define a cluster-receiver channel at the local queue manager. You do not need to define a transmission queue or a remote queue object. For information, see Clusters.

More about name resolution

The effect of the remote queue definition is to define a physical destination queue name and queue manager name. These names are put in the transmission headers of messages.

Incoming messages from an adjacent system have already had this type of name resolution carried out by the original queue manager. Therefore they have the transmission header showing the physical destination queue name and queue manager name. These messages are unaffected by remote queue definitions.

Choosing the transmission queue

You can use a remote queue definition to allow a different transmission queue to send messages to the same adjacent queue manager.

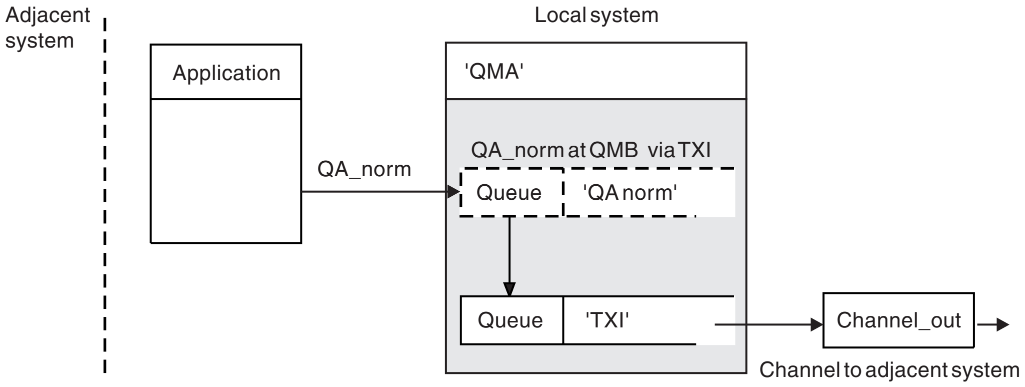


Figure 110. The remote queue definition allows a different transmission queue to be used

In a distributed-queuing environment, when you need to change a message flow from one channel to another, use the same system configuration as shown in Figure 109 on page 763 in "Putting messages on remote queues" on page 762. Figure 110 in this topic shows how you use the remote queue definition to send messages over a different transmission queue, and therefore over a different channel, to the same adjacent queue manager.

For the configuration shown in Figure 110, you must provide the remote queue object 'QA_norm', and the transmission queue 'TX1'. You must provide 'QA_norm' to choose the 'QA_norm' queue at the remote queue manager, the transmission queue 'TX1', and the queue manager 'QMB_priority'. Specify 'TX1' in the definition of the channel adjacent to the system.

Messages are placed on transmission queue 'TX1' with a transmission header containing 'QA_norm at QMB_priority', and are sent over the channel to the adjacent system.

The channel_back has been left out of this illustration because it would need a queue manager alias.

In a clustering environment, you do not need to define a transmission queue or a remote queue definition. For more information, see "Cluster queues" on page 892.

Receiving messages

You can configure the queue manager to receive messages from other queue managers. You must ensure that unintentional name resolution does not occur.

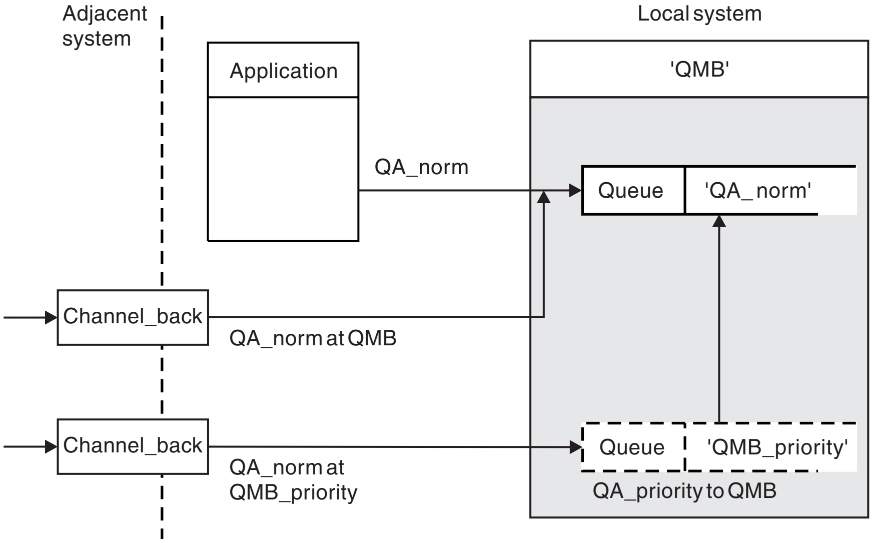


Figure 111. Receiving messages directly, and resolving alias queue manager name

As well as arranging for messages to be sent, the system administrator must also arrange for messages to be received from adjacent queue managers. Received messages contain the physical name of the destination queue manager and queue in the transmission header. They are treated the same as messages from a local application that specifies both queue manager name and queue name. Because of this treatment, you need to ensure that messages entering your system do not have an unintentional name resolution carried out. See Figure 111 for this scenario.

For this configuration, you must prepare:

- Message channels to receive messages from adjacent queue managers
- A queue manager alias definition to resolve an incoming message flow, 'QMB_priority', to the local queue manager name, 'QMB'
- The local queue, 'QA_norm', if it does not exist

Receiving alias queue manager names

The use of the queue manager alias definition in this illustration has not selected a different destination queue manager. Messages passing through this local queue manager and addressed to 'QMB_priority' are intended for queue manager 'QMB'. The alias queue manager name is used to create the separate message flow.

Passing messages through your system

You can pass messages through your system in three ways - using the location name, using an alias for the queue manager, or selecting a transmission queue.

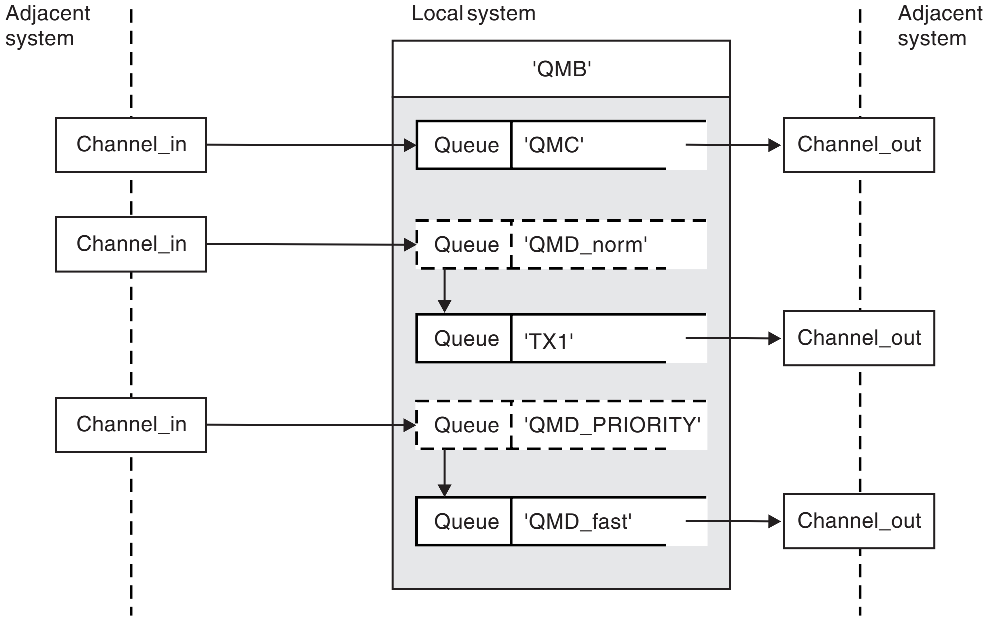


Figure 112. Three methods of passing messages through your system

The technique shown in Figure 111 on page 765 in "Receiving messages" on page 764, showed how an alias flow is captured. Figure 112 illustrates the ways networks are built up by bringing together the techniques previously described.

The configuration shows a channel delivering three messages with different destinations:

1. `QB` at `QMC`
2. `QB` at `QMD_norm`
3. `QB` at `QMD_PRIORITY`

You must pass the first message flow through your system unchanged. You must pass the second message flow through a different transmission queue and channel. For the second message flow you must also resolve messages for the alias queue manager name `QMD_norm` to the queue manager `QMD`. The third message flow chooses a different transmission queue without any other change.

In a clustering environment, messages are passed through a cluster transmission queue. Normally a single transmission queue, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, transfers all messages to all queue managers in all clusters that the queue manager is a member of; see A cluster of queue managers. You can define separate transmission queues for all or some of the queue managers in the clusters that the queue manager is a member of.

The following methods describe techniques applicable to a distributed-queuing environment.

Use these methods

For these configurations, you must prepare the:

- Input channel definitions
- Output channel definitions
- Transmission queues:
  - `QMC`

  - TX1
  - QMD_fast
- Queue manager alias definitions:
  - QMD_norm with QMD_norm to QMD through TX1
  - QMD_PRIORITY with QMD_PRIORITY to QMD_PRIORITY through QMD_fast

Note: None of the message flows shown in the example changes the destination queue. The queue manager name aliases provide separation of message flows.

Method 1: Use the incoming location name

You are going to receive messages with a transmission header containing another location name, such as QMC. The simplest configuration is to create a transmission queue with that name, QMC. The channel that services the transmission queue delivers the message unchanged to the next destination.

Method 2: Use an alias for the queue manager

The second method is to use the queue manager alias object definition, but specify a new location name, QMD, and a particular transmission queue, TX1. This action:

- Terminates the alias message flow setup by the queue manager name alias QMD_norm, that is, the named class of service QMD_norm.
- Changes the transmission headers on these messages from QMD_norm to QMD.

Method 3: Select a transmission queue

The third method is to have a queue manager alias object defined with the same name as the destination location, QMD_PRIORITY. Use the queue manager alias definition to select a particular transmission queue, QMD_fast, and therefore another channel. The transmission headers on these messages remain unchanged.

Separating message flows

You can use a queue manager alias to create separate message flows to send messages to the same queue manager.

In a distributed-queuing environment, the need to separate messages to the same queue manager into different message flows can arise for a number of reasons. For example:

- You might need to provide a separate flow for large, medium, and small messages. This need also applies in a clustering environment and, in this case, you can create clusters that overlap. There are a number of reasons you might do so, for example:
  - To allow different organizations to have their own administration.
  - To allow independent applications to be administered separately.
  - To create a class of service. For example, you could have a cluster called STAFF that is a subset of the cluster called STUDENTS. When you put a message to a queue advertised in the STAFF cluster, a restricted channel is used. When you put a message to a queue advertised in the STUDENTS cluster, either a general channel or a restricted channel can be used.
  - To create test and production environments.
- It might be necessary to route incoming messages by different paths from the path of the locally generated messages.
- Your installation might require to schedule the movement of messages at certain times (for example, overnight) and the messages then need to be stored in reserved queues until scheduled.

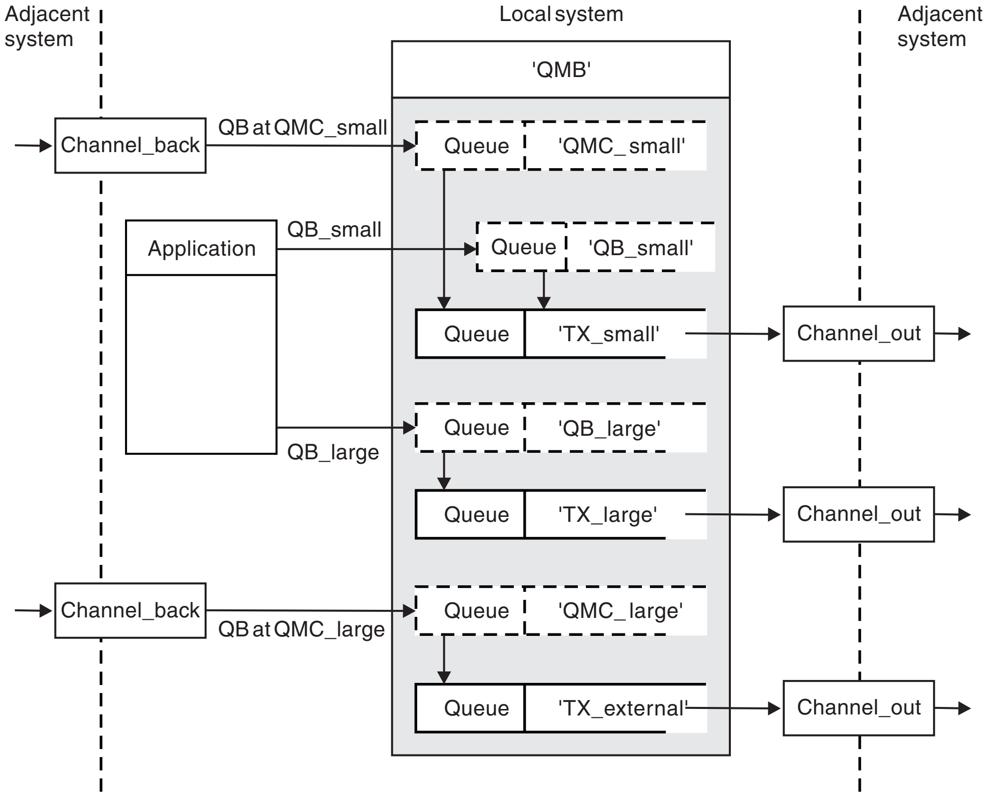


Figure 113. Separating messages flows

In the example shown in Figure 113, the two incoming flows are to alias queue manager names 'QMC_small' and 'QMC_large'. You provide these flows with a queue manager alias definition to capture these flows for the local queue manager. You have an application addressing two remote queues and you need these message flows to be kept separate. You provide two remote queue definitions that specify the same location, 'QMC', but specify different transmission queues. This definition keeps the flows separate, and nothing extra is needed at the far end as they have the same destination queue manager name in the transmission headers. You provide:

- The incoming channel definitions
- The two remote queue definitions QB_small and QB_large
- The two queue manager alias definitions QMC_small and QMC_large
- The three sending channel definitions
- Three transmission queues: TX_small, TX_large, and TX_external

Coordination with adjacent systems

When you use a queue manager alias to create a separate message flow, you need to coordinate this activity with the system administrator at the remote end of the message channel to ensure that the corresponding queue manager alias is available there.

Concentrating messages to diverse locations

You can concentrate messages destined for various locations on to a single channel.

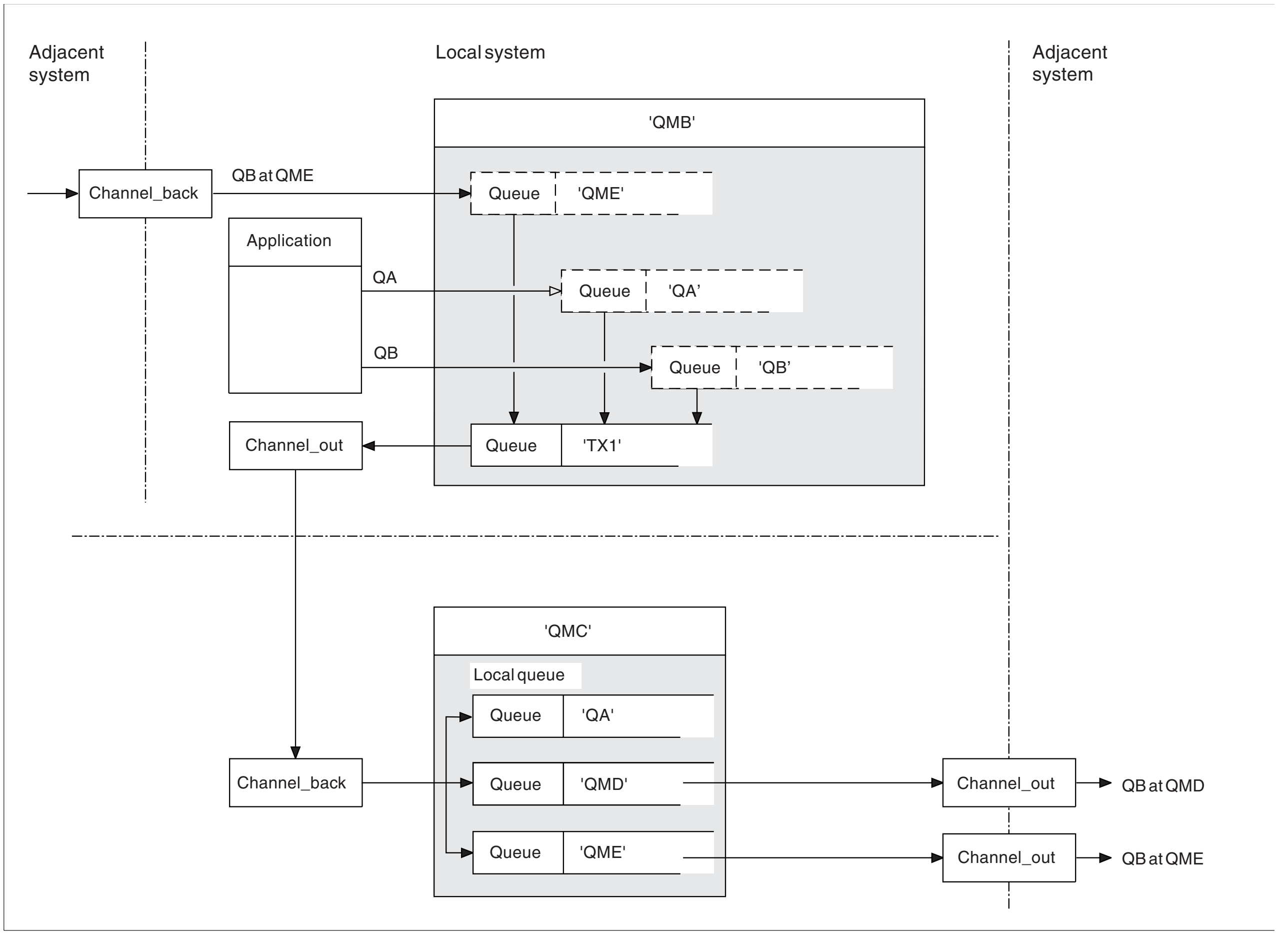


Figure 114. Combining message flows on to a channel

Figure 114 illustrates a distributed-queuing technique for concentrating messages that are destined for various locations on to one channel. Two possible uses would be:

- Concentrating message traffic through a gateway
- Using wide bandwidth highways between nodes

In this example, messages from different sources, local and adjacent, and having different destination queues and queue managers, are flowed through transmission queue 'TX1' to queue manager QMC. Queue manager QMC delivers the messages according to the destinations. One set to a transmission queue 'QMD' for onward transmission to queue manager QMD. Another set to a transmission queue 'QME' for onward transmission to queue manager QME. Other messages are put on the local queue 'QA'.

You must provide:

- Channel definitions
- Transmission queue TX1
- Remote queue definitions:
  - QA with 'QA at QMC through TX1'
  - QB with 'QB at QMD through TX1'
- Queue manager alias definition:

- QME with 'QME through TX1'

The complementary administrator who is configuring QMC must provide:

- Receiving channel definition with the same channel name
- Transmission queue QMD with associated sending channel definition
- Transmission queue QME with associated sending channel definition
- Local queue object QA.

Diverting message flows to another destination

You can redefine the destination of certain messages using queue manager aliases and transmission queues.

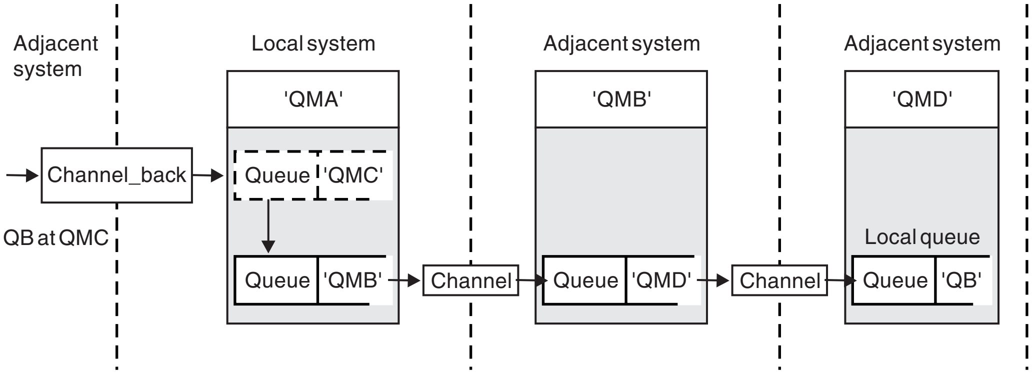


Figure 115. Diverting message streams to another destination

Figure 115 illustrates how you can redefine the destination of certain messages. Incoming messages to QMA are destined for 'QB at QMC'. They normally arrive at QMA and be placed on a transmission queue called QMC which has been part of a channel to QMC. QMA must divert the messages to QMD, but is able to reach QMD only over QMB. This method is useful when you need to move a service from one location to another, and allow subscribers to continue to send messages on a temporary basis until they have adjusted to the new address.

The method of rerouting incoming messages destined for a certain queue manager to a different queue manager uses:

- A queue manager alias to change the destination queue manager to another queue manager, and to select a transmission queue to the adjacent system
- A transmission queue to serve the adjacent queue manager
- A transmission queue at the adjacent queue manager for onward routing to the destination queue manager

You must provide:

- Channel_back definition
- Queue manager alias object definition QMC with QB at QMD through QMB
- Channel_out definition
- The associated transmission queue QMB

The complementary administrator who is configuring QMB must provide:

- The corresponding channel_back definition
- The transmission queue, QMD

- The associated channel definition to QMD

You can use aliases within a clustering environment. For information, see “Queue-manager aliases and clusters” on page 985.

Sending messages to a distribution list

You can use a single MQPUT call to have an application send a message to several destinations.

In WebSphere MQ on all platforms except z/OS, an application can send a message to several destinations with a single MQPUT call. You can do so in both a distributed-queuing environment and a clustering environment. You have to define the destinations in a distribution list, as described in Distribution lists.

Not all queue managers support distribution lists. When an MCA establishes a connection with a partner, it determines whether the partner supports distribution lists and sets a flag on the transmission queue accordingly. If an application tries to send a message that is destined for a distribution list but the partner does not support distribution lists, the sending MCA intercepts the message and puts it onto the transmission queue once for each intended destination.

A receiving MCA ensures that messages sent to a distribution list are safely received at all the intended destinations. If any destinations fail, the MCA establishes which ones have failed. It then can generate exception reports for them and can try to send the messages to them again.

Reply-to queue

You can create a complete remote queue processing loop using a reply-to queue.

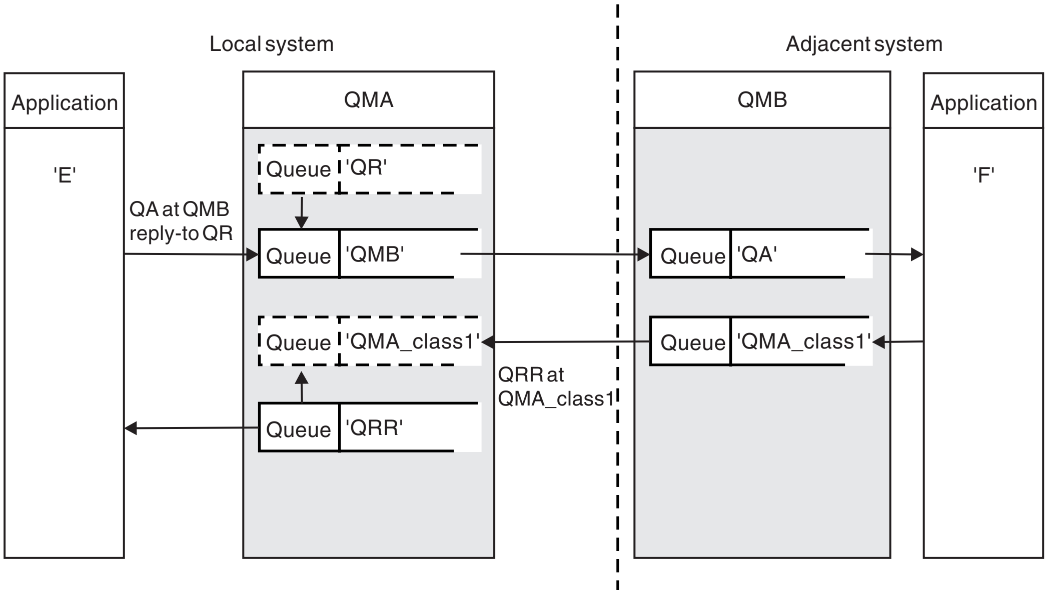


Figure 116. Reply-to queue name substitution during PUT call

A complete remote queue processing loop using a reply-to queue is shown in Figure 116. This loop applies in both a distributed-queuing environment and a clustering environment. The details are as shown in Table 109 on page 779.

The application opens QA at QMB and puts messages on that queue. The messages are given a reply-to queue name of QR, without the queue manager name being specified. Queue manager QMA finds the reply-to queue object QR and extracts from it the alias name of QRR and the queue manager name QMA_class1. These names are put into the reply-to fields of the messages.

Reply messages from applications at QMB are addressed to QRR at QMA_class1. The queue manager alias name definition QMA_class1 is used by the queue manager to flow the messages to itself, and to queue QRR.

This scenario depicts the way you give applications the facility to choose a class of service for reply messages. The class is implemented by the transmission queue QMA_class1 at QMB, together with the queue manager alias definition, QMA_class1 at QMA. In this way, you can change an application's reply-to queue so that the flows are segregated without involving the application. The application always chooses QR for this particular class of service. You have the opportunity to change the class of service with the reply-to queue definition QR.

You must create:

- Reply-to queue definition QR
- Transmission queue object QMB
- Channel_out definition
- Channel_back definition
- Queue manager alias definition QMA_class1
- Local queue object QRR, if it does not exist

The complementary administrator at the adjacent system must create:

- Receiving channel definition
- Transmission queue object QMA_class1
- Associated sending channel
- Local queue object QA.

Your application programs use:

- Reply-to queue name QR in put calls
- Queue name QRR in get calls

In this way, you can change the class of service as necessary, without involving the application. You change the reply-to alias 'QR', together with the transmission queue 'QMA_class1' and queue manager alias 'QMA_class1'.

If no reply-to alias object is found when the message is put on the queue, the local queue manager name is inserted in the blank reply-to queue manager name field. The reply-to queue name remains unchanged.

Name resolution restriction

Because the name resolution has been carried out for the reply-to queue at 'QMA' when the original message was put, no further name resolution is allowed at 'QMB'. The message is put with the physical name of the reply-to queue by the replying application.

The applications must be aware that the name they use for the reply-to queue is different from the name of the actual queue where the return messages are to be found.

For example, when two classes of service are provided for the use of applications with reply-to queue alias names of 'C1_alias', and 'C2_alias', the applications use these names as reply-to queue names in the message put calls. However, the applications actually expect messages to appear in queues 'C1' for 'C1_alias' and 'C2' for 'C2_alias'.

However, an application is able to make an inquiry call on the reply-to alias queue to check for itself the name of the real queue it must use to get the reply messages.

Related concepts:

"How to create queue manager and reply-to aliases" on page 761
This topic explains the three ways that you can create a remote queue definition.

"Reply-to queue alias example"
This example illustrates the use of a reply-to alias to select a different route (transmission queue) for returned messages. The use of this facility requires the reply-to queue name to be changed in cooperation with the applications.

"How the example works" on page 775
An explanation of the example and how the queue manager uses the reply-to queue alias.

"Reply-to queue alias walk-through" on page 775
A walk-through of the process from an application putting a message on a remote queue through to the same application removing the reply message from the alias reply-to queue.

Reply-to queue alias example:

This example illustrates the use of a reply-to alias to select a different route (transmission queue) for returned messages. The use of this facility requires the reply-to queue name to be changed in cooperation with the applications.

As shown in Figure 117, the return route must be available for the reply messages, including the transmission queue, channel, and queue manager alias.

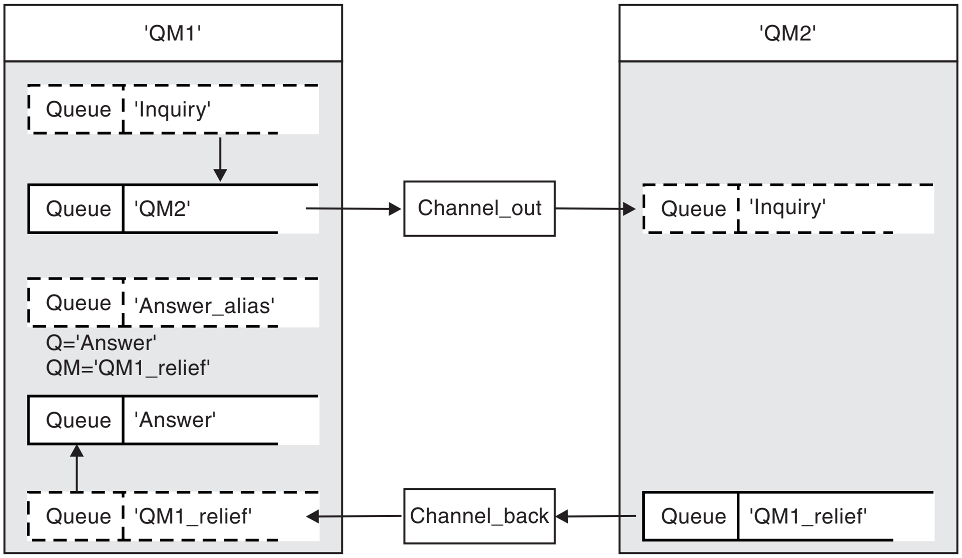


Figure 117. Reply-to queue alias example

This example is for requester applications at 'QM1' that send messages to server applications at 'QM2'. The messages on the server are to be returned through an alternative channel using transmission queue 'QM1_relief' (the default return channel would be served with a transmission queue 'QM1').

The reply-to queue alias is a particular use of the remote queue definition named 'Answer_alias'. Applications at QM1 include this name, 'Answer_alias', in the reply-to field of all messages that they put on queue 'Inquiry'.

Reply-to queue definition 'Answer_alias' is defined as 'Answer at QM1_relief'. Applications at QM1 expect their replies to appear in the local queue named 'Answer'.

Server applications at QM2 use the reply-to field of received messages to obtain the queue and queue manager names for the reply messages to the requester at QM1.

Definitions used in this example at QM1

The WebSphere MQ system administrator at QM1 must ensure that the reply-to queue 'Answer' is created along with the other objects. The name of the queue manager alias, marked with a '*', must agree with the queue manager name in the reply-to queue alias definition, also marked with an '*'.

Object	Definition	
Local transmission queue	QM2	
Remote queue definition	Object name	Inquiry
	Remote queue manager name	QM2
	Remote queue name	Inquiry
	Transmission queue name	QM2 (DEFAULT)
Queue manager alias	Object name	QM1_relief *
	Queue manager name	QM1
	Queue name	(blank)
Reply-to queue alias	Object name	Answer_alias
	Remote queue manager name	QM1_relief *
	Remote queue name	Answer

Put definition at QM1

Applications fill the reply-to fields with the reply-to queue alias name, and leave the queue manager name field blank.

Field	Content
Queue name	Inquiry
Queue manager name	(blank)
Reply-to queue name	Answer_alias
Reply-to queue manager	(blank)

Definitions used in this example at QM2

The WebSphere MQ system administrator at QM2 must ensure that the local queue exists for the incoming messages, and that the correctly named transmission queue is available for the reply messages.

Object	Definition
Local queue	Inquiry
Transmission queue	QM1_relief

Put definition at QM2

Applications at QM2 retrieve the reply-to queue name and queue manager name from the original message and use them when putting the reply message on the reply-to queue.

Field	Content
Queue name	Answer
Queue manager name	QM1_relief

How the example works:

An explanation of the example and how the queue manager uses the reply-to queue alias.

In this example, requester applications at QM1 always use 'Answer_alias' as the reply-to queue in the relevant field of the put call. They always retrieve their messages from the queue named 'Answer'.

The reply-to queue alias definitions are available for use by the QM1 system administrator to change the name of the reply-to queue 'Answer', and of the return route 'QM1_relief'.

Changing the queue name 'Answer' is normally not useful because the QM1 applications are expecting their answers in this queue. However, the QM1 system administrator is able to change the return route (class of service), as necessary.

How the queue manager uses the reply-to queue alias

Queue manager QM1 retrieves the definitions from the reply-to queue alias when the reply-to queue name, included in the put call by the application, is the same as the reply-to queue alias, and the queue manager part is blank.

The queue manager replaces the reply-to queue name in the put call with the queue name from the definition. It replaces the blank queue manager name in the put call with the queue manager name from the definition.

These names are carried with the message in the message descriptor.

Table 106. Reply-to queue alias

Field name	Put call	Transmission header
Reply-to queue name	Answer_alias	Answer
Reply-to queue manager name	(blank)	QM1_relief

Reply-to queue alias walk-through:

A walk-through of the process from an application putting a message on a remote queue through to the same application removing the reply message from the alias reply-to queue.

To complete this example, let us look at the process.

1. The application opens a queue named 'Inquiry', and puts messages to it. The application sets the reply-to fields of the message descriptor to:

Reply-to queue name	**Answer_alias**
Reply-to queue manager name	(blank)

2. Queue manager 'QM1' responds to the blank queue manager name by checking for a remote queue definition with the name 'Answer_alias'. If none is found, the queue manager places its own name, 'QM1', in the reply-to queue manager field of the message descriptor.
3. If the queue manager finds a remote queue definition with the name 'Answer_alias', it extracts the queue name and queue manager names from the definition (queue name='Answer' and queue manager name= 'QM1_relief'). It then puts them into the reply-to fields of the message descriptor.
4. The queue manager 'QM1' uses the remote queue definition 'Inquiry' to determine that the intended destination queue is at queue manager 'QM2', and the message is placed on the transmission queue 'QM2'. 'QM2' is the default transmission queue name for messages destined for queues at queue manager 'QM2'.
5. When queue manager 'QM1' puts the message on the transmission queue, it adds a transmission header to the message. This header contains the name of the destination queue, 'Inquiry', and the destination queue manager, 'QM2'.
6. The message arrives at queue manager 'QM2', and is placed on the 'Inquiry' local queue.
7. An application gets the message from this queue and processes the message. The application prepares a reply message, and puts this reply message on the reply-to queue name from the message descriptor of the original message:

Reply-to queue name	**Answer**
Reply-to queue manager name	QM1_relief

8. Queue manager 'QM2' carries out the put command. Finding that the queue manager name, 'QM1_relief', is a remote queue manager, it places the message on the transmission queue with the same name, 'QM1_relief'. The message is given a transmission header containing the name of the destination queue, 'Answer', and the destination queue manager, 'QM1_relief'.
9. The message is transferred to queue manager 'QM1'. The queue manager, recognizes that the queue manager name 'QM1_relief' is an alias, extracts from the alias definition 'QM1_relief' the physical queue manager name 'QM1'.
10. Queue manager 'QM1' then puts the message on the queue name contained in the transmission header, 'Answer'.
11. The application extracts its reply message from the queue 'Answer'.

Networking considerations

In a distributed-queuing environment, because message destinations are addressed with just a queue name and a queue manager name, certain rules apply.

1. Where the queue manager name is given, and the name is different from the local queue manager name:
   - A transmission queue must be available with the same name. This transmission queue must be part of a message channel moving messages to another queue manager, or
   - A queue manager alias definition must exist to resolve the queue manager name to the same, or another queue manager name, and optional transmission queue, or
   - If the transmission queue name cannot be resolved, and a default transmission queue has been defined, the default transmission queue is used.
2. Where only the queue name is supplied, a queue of any type but with the same name must be available on the local queue manager. This queue can be a remote queue definition which resolves to: a transmission queue to an adjacent queue manager, a queue manager name, and an optional transmission queue.

To see how this works in a clustering environment, see the appropriate topics in the How clusters work section of the product documentation.

Consider the scenario of a message channel moving messages from one queue manager to another in a distributed-queuing environment.

The messages being moved have originated from any other queue manager in the network, and some messages might arrive that have an unknown queue manager name as destination. This issue can occur when a queue manager name has changed or has been removed from the system, for example.

The channel program recognizes this situation when it cannot find a transmission queue for these messages, and places the messages on your undelivered-message (dead-letter) queue. It is your responsibility to look for these messages and arrange for them to be forwarded to the correct destination. Alternatively, return them to the originator, where the originator can be ascertained.

Exception reports are generated in these circumstances, if report messages were requested in the original message.

Name resolution convention

Name resolution that changes the identity of the destination queue (that is, logical to physical name changing), only occurs once, and only at the originating queue manager.

Subsequent use of the various alias possibilities must only be used when separating and combining message flows.

Return routing

Messages can contain a return address in the form of the name of a queue and queue manager. This return address form can be used in both a distributed-queuing environment and a clustering environment.

This address is normally specified by the application that creates the message. It can be modified by any application that then handles the message, including user exit applications.

Irrespective of the source of this address, any application handling the message might choose to use this address for returning answer, status, or report messages to the originating application.

The way these response messages are routed is not different from the way the original message is routed. You need to be aware that the message flows you create to other queue managers need corresponding return flows.

Physical name conflicts

The destination reply-to queue name has been resolved to a physical queue name at the original queue manager. It must not be resolved again at the responding queue manager.

It is a likely possibility for name conflict problems that can only be prevented by a network-wide agreement on physical and logical queue names.

Managing queue name translations

When you create a queue manager alias definition or a remote queue definition, the name resolution is carried out for every message carrying that name. This situation must be managed.

This description is provided for application designers and channel planners concerned with an individual system that has message channels to adjacent systems. It takes a local view of channel planning and control.

When you create a queue manager alias definition or a remote queue definition, the name resolution is carried out for every message carrying that name, regardless of the source of the message. To oversee this situation, which might involve large numbers of queues in a queue manager network, you keep tables of:

- The names of source queues and of source queue managers with respect to resolved queue names, resolved queue manager names, and resolved transmission queue names, with method of resolution
- The names of source queues with respect to:
  - Resolved destination queue names
  - Resolved destination queue manager names
  - Transmission queues
  - Message channel names
  - Adjacent system names
  - Reply-to queue names

Note: The use of the term *source* in this context refers to the queue name or the queue manager name provided by the application, or a channel program when opening a queue for putting messages.

An example of each of these tables is shown in Table 107, Table 108, and Table 109 on page 779.

The names in these tables are derived from the examples in this section, and this table is not intended as a practical example of queue name resolution in one node.

Table 107. Queue name resolution at queue manager QMA

Source queue specified when queue is opened	Source queue manager specified when queue is opened	Resolved queue name	Resolved queue manager name	Resolved transmission queue name	Resolution type
QA_norm	-	QA_norm	QMB	QMB	Remote queue
(any)	QMB	-	-	QMB	(none)
QA_norm	-	QA_norm	QMB	TX1	Remote queue
QB	QMC	QB	QMD	QMB	Queue manager alias

Table 108. Queue name resolution at queue manager QMB

Source queue specified when queue is opened	Source queue manager specified when queue is opened	Resolved queue name	Resolved queue manager name	Resolved transmission queue name	Resolution type
QA_norm	-	QA_norm	QMB	-	(none)
QA_norm	QMB	QA_norm	QMB	-	(none)
QA_norm	QMB_PRIORITY	QA_norm	QMB	-	Queue manager alias
(any)	QMC	(any)	QMC	QMC	(none)
(any)	QMD_norm	(any)	QMD_norm	TX1	Queue manager alias

Table 108. Queue name resolution at queue manager QMB (continued)

Source queue specified when queue is opened	Source queue manager specified when queue is opened	Resolved queue name	Resolved queue manager name	Resolved transmission queue name	Resolution type
(any)	QMD_PRIORITY	(any)	QMD_PRIORITY	QMD_fast	Queue manager alias
(any)	QMC_small	(any)	QMC_small	TX_small	Queue manager alias
(any)	QMC_large	(any)	QMC_large	TX_external	Queue manager alias
QB_small	QMC	QB_small	QMC	TX_small	Remote queue
QB_large	QMC	QB_large	QMC	TX_large	Remote queue
(any)	QME	(any)	QME	TX1	Queue manager alias
QA	QMC	QA	QMC	TX1	Remote queue
QB	QMD	QB	QMD	TX1	Remote queue

Table 109. Reply-to queue name translation at queue manager QMA

Application design		Reply-to alias definition	
Local QMGR	**Queue name for messages**	**Reply-to queue alias name**	**Redefined to**
QMA	QRR	QR	QRR at QMA_class1

Channel message sequence numbering

The channel uses sequence numbers to assure that messages are delivered, delivered without duplication, and stored in the same order as they were taken from the transmission queue.

The sequence number is generated at the sending end of the channel and is incremented by one before being used, which means that the current sequence number is the number of the last message sent. This information can be displayed using DISPLAY CHSTATUS (see MQSC reference). The sequence number and an identifier called the LUWID are stored in persistent storage for the last message transferred in a batch. These values are used during channel start-up to ensure that both ends of the link agree on which messages have been transferred successfully.

Sequential retrieval of messages

If an application puts a sequence of messages to the same destination queue, those messages can be retrieved in sequence by a ***single*** application with a sequence of MQGET operations, if the following conditions are met:

- All the put requests were done from the same application.
- All the put requests were either from the same unit of work, or all the put requests were made outside of a unit of work.
- The messages all have the same priority.
- The messages all have the same persistence.
- For remote queuing, the configuration is such that there can only be one path from the application making the put request, through its queue manager, through intercommunication, to the destination queue manager and the target queue.
- The messages are not put to a dead-letter queue (for example, if a queue is temporarily full).
- The application getting the message does not deliberately change the order of retrieval, for example by specifying a particular *MsgId* or *CorrelId* or by using message priorities.
- Only one application is doing get operations to retrieve the messages from the destination queue. If there is more than one application, these applications must be designed to get all the messages in each sequence put by a sending application.

Note: Messages from other tasks and units of work might be interspersed with the sequence, even where the sequence was put from within a single unit of work.

If these conditions cannot be met, and the order of messages on the target queue is important, then the application can be coded to use its own message sequence number as part of the message to assure the order of the messages.

Sequence of retrieval of fast, nonpersistent messages

Nonpersistent messages on a fast channel might overtake persistent messages on the same channel and so arrive out of sequence. The receiving MCA puts the nonpersistent messages on the destination queue immediately and makes them visible. Persistent messages are not made visible until the next sync point.

Loopback testing

Loopback testing is a technique on non-z/OS platforms that allows you to test a communications link without actually linking to another machine.

You set up a connection between two queue managers as though they are on separate machines, but you test the connection by looping back to another process on the same machine. This technique means that you can test your communications code without requiring an active network.

The way you do so depends on which products and protocols you are using.

On Windows systems, you can use the "loopback" adapter.

Refer to the documentation for the products you are using for more information.

Route tracing and activity recording

You can confirm the route a message takes through a series of queue managers in two ways.

You can use the WebSphere MQ display route application, available through the control command **dspmqrte**, or you can use activity recording. Both of these topics are described in Monitoring reference.

Introduction to distributed queue management

Distributed queue management (DQM) is used to define and control communication between queue managers.

Distributed queue management:

- Enables you to define and control communication channels between queue managers
- Provides you with a message channel service to move messages from a type of *local queue*, known as a transmission queue, to communication links on a local system, and from communication links to local queues at a destination queue manager
- Provides you with facilities for monitoring the operation of channels and diagnosing problems, using panels, commands, and programs

Channel definitions associate channel names with transmission queues, communication link identifiers, and channel attributes. Channel definitions are implemented in different ways on different platforms. Message sending and receiving is controlled by programs known as *message channel agents* (MCAs), which use the channel definitions to start up and control communication.

The MCAs in turn are controlled by DQM itself. The structure is platform-dependent, but typically includes listeners and trigger monitors, together with operator commands and panels.

A *message channel* is a one-way pipe for moving messages from one queue manager to another. Thus a message channel has two end-points, represented by a pair of MCAs. Each end point has a definition of its end of the message channel. For example, one end would define a sender, the other end a receiver.

For details of how to define channels, see:

- Windows UNIX Linux "Monitoring and controlling channels on Windows, UNIX and Linux platforms" on page 807

For message channel planning examples, see:

- Windows UNIX Linux Message channel planning example for distributed platforms

For information about channel exits, see Channel-exit programs for messaging channels.

Related concepts:

"Message sending and receiving"
The following figure shows the distributed queue management model, detailing the relationships between entities when messages are transmitted. It also shows the flow for control.

"Channel control function" on page 787
The channel control function provides facilities for you to define, monitor, and control channels.

"What happens when a message cannot be delivered?" on page 799
When a message cannot be delivered, the MCA can process it in several ways. It can try again, it can return-to-sender, or it can put it on the dead-letter queue.

"Initialization and configuration files" on page 803
The handling of channel initialization data depends on your WebSphere MQ platform.

"Data conversion" on page 805
WebSphere MQ messages might require data conversion when sent between queues on different queue managers.

"Writing your own message channel agents" on page 805
WebSphere MQ allows you to write your own message channel agent (MCA) programs or to install one from an independent software vendor.

"Other things to consider for distributed queue management" on page 806
Other topics to consider when preparing WebSphere MQ for distributed queue management. This topic covers Undelivered-message queue, Queues in use, System extensions and user-exit programs, and Running channels and listeners as trusted applications.

Related information:

Example configuration information

Message sending and receiving

The following figure shows the distributed queue management model, detailing the relationships between entities when messages are transmitted. It also shows the flow for control.

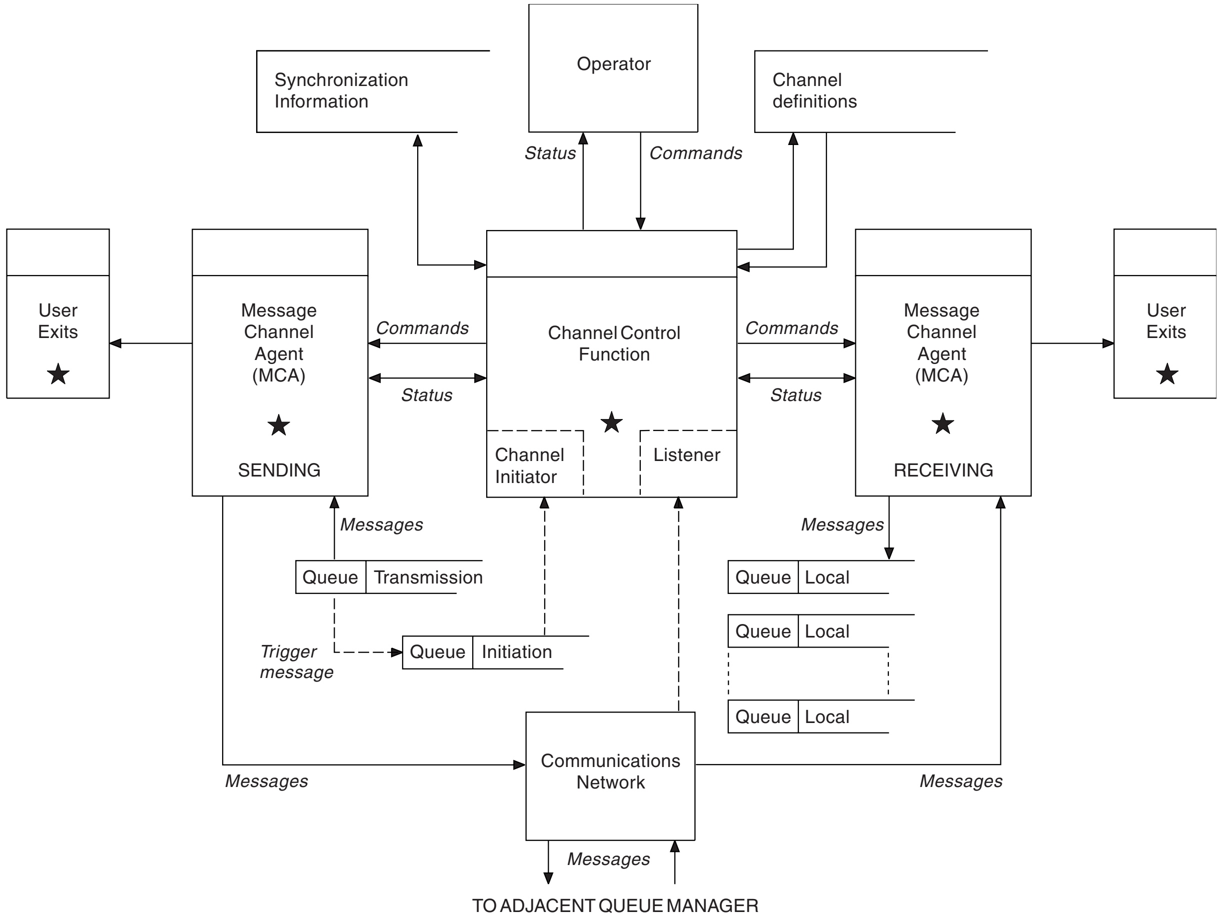


Figure 118. Distributed queue management model

Note:

1. There is one MCA per channel, depending on the platform. There might be one or more channel control functions for a particular queue manager.
2. The implementation of MCAs and channel control functions is highly platform-dependent. They can be programs or processes or threads, and they can be a single entity or many comprising several independent or linked parts.
3. All components marked with a star can use the MQI.

Channel parameters

An MCA receives its parameters in one of several ways:

- If started by a command, the channel name is passed in a data area. The MCA then reads the channel definition directly to obtain its attributes.
- For sender, and in some cases server channels, the MCA can be started automatically by the queue manager trigger. The channel name is retrieved from the trigger process definition, where applicable, and is passed to the MCA. The remaining processing is the same as previously described. Server channels must only be set up to trigger if they are fully qualified, that is, they specify a CONNAME to connect to.
- If started remotely by a sender, server, requester, or client-connection, the channel name is passed in the initial data from the partner message channel agent. The MCA reads the channel definition directly to obtain its attributes.

Certain attributes not defined in the channel definition are also negotiable:

Split messages
If one end does not support split messages then the split messages are not sent.

Conversion capability
If one end cannot perform the necessary code page conversion or numeric encoding conversion when needed, the other end must handle it. If neither end supports it, when needed, the channel cannot start.

Distribution list support
If one end does not support distribution lists, the partner MCA sets a flag in its transmission queue so that it knows to intercept messages intended for multiple destinations.

Channel status and sequence numbers

Message channel agent programs keep records of the current sequence number and logical unit of work number for each channel, and of the general status of the channel. Some platforms allow you to display this status information to help you control channels.

How to send a message to another queue manager

This section describes the simplest way to send a message between queue managers, including prerequisites and authorizations required. Other methods can also be used to send messages to a remote queue manager.

Before you send a message from one queue manager to another, you need to do the following steps:

1. Check that your chosen communication protocol is available.
2. Start the queue managers.
3. Start the channel initiators.
4. Start the listeners.

You also need to have the correct WebSphere MQ security authorization to create the objects required.

To send messages from one queue manager to another:

- Define the following objects on the source queue manager:
  - Sender channel
  - Remote queue definition
  - Initiation queue (optional)
  - Transmission queue
  - Dead-letter queue
- Define the following objects on the target queue manager:
  - Receiver channel
  - Target queue
  - Dead-letter queue

You can use several different methods to define these objects, depending on your WebSphere MQ platform:

- On all platforms, you can use the WebSphere MQ script commands (MQSC) described in The MQSC commands the programmable command format (PCF) commands described in Automating administration tasks, or the WebSphere MQ Explorer.

See the following subtopics for more information on creating the components for sending messages to another queue manager:

Related concepts:

"Creating and managing queue managers" on page 751
Before you can use messages and queues, you must create and start at least one queue manager and its associated objects.

"WebSphere MQ distributed-messaging techniques" on page 760
The subtopics in this section describe techniques that are of use when planning channels. These subtopics describe techniques to help you plan how to connect your queue managers together, and manage the flow of messages between your applications.

"Introduction to distributed queue management" on page 780
Distributed queue management (DQM) is used to define and control communication between queue managers.

"Triggering channels" on page 801
WebSphere MQ provides a facility for starting an application automatically when certain conditions on a queue are met. This facility is called triggering.

"Safety of messages" on page 799
In addition to the typical recovery features of WebSphere MQ, distributed queue management ensures that messages are delivered properly by using a sync point procedure coordinated between the two ends of the message channel. If this procedure detects an error, it closes the channel so that you can investigate the problem, and keeps the messages safely in the transmission queue until the channel is restarted.

"Monitoring and controlling channels on Windows, UNIX and Linux platforms" on page 807
For DQM you need to create, monitor, and control the channels to remote queue managers. You can control channels using commands, programs, WebSphere MQ Explorer, files for the channel definitions, and a storage area for synchronization information.

"Configuring connections between the server and client" on page 832
To configure the communication links between WebSphere MQ MQI clients and servers, decide on your communication protocol, define the connections at both ends of the link, start a listener, and define channels.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

Defining the channels:

To send messages from one queue manager to another, you must define two channels. You must define one channel on the source queue manager and one channel on the target queue manager.

On the source queue manager

Define a channel with a channel type of SENDER. You need to specify the following:

- The name of the transmission queue to be used (the XMITQ attribute).
- The connection name of the partner system (the CONNAME attribute).
- The name of the communication protocol you are using (the TRPTYPE attribute). On WebSphere MQ for z/OS, the protocol must be TCP or LU6.2. On other platforms, you do not have to specify this. You can leave it to pick up the value from your default channel definition.

Details of all the channel attributes are given in Channel attributes.

On the target queue manager

Define a channel with a channel type of RECEIVER, and the same name as the sender channel.

Specify the name of the communication protocol you are using (the TRPTYPE attribute). On WebSphere MQ for z/OS, the protocol must be TCP or LU6.2. On other platforms, you do not have to specify this. You can leave it to pick up the value from your default channel definition.

Receiver channel definitions can be generic. This means that if you have several queue managers communicating with the same receiver, the sending channels can all specify the same name for the receiver, and one receiver definition applies to them all.

When you have defined the channel, you can test it using the PING CHANNEL command. This command sends a special message from the sender channel to the receiver channel and checks that it is returned.

Defining the queues:

To send messages from one queue manager to another, you must define up to six queues. You must define up to four queues on the source queue manager, and up to two queues on the target queue manager.

On the source queue manager

- Remote queue definition

  In this definition, specify the following:

  Remote queue manager name
  The name of the target queue manager.

  Remote queue name
  The name of the target queue on the target queue manager.

  Transmission queue name
  The name of the transmission queue. You do not have to specify this transmission queue name. If you do not, a transmission queue with the same name as the target queue manager is used. If this does not exist, the default transmission queue is used. You are advised to give the transmission queue the same name as the target queue manager so that the queue is found by default.
- Initiation queue definition

  Required on z/OS, and optional on other platforms. Consider naming the initiation queue SYSTEM.CHANNEL.INITQ. on other platforms.
- Transmission queue definition

  A local queue with the USAGE attribute set to XMITQ.
- Dead-letter queue definition

  Define a dead-letter queue to which undelivered messages can be written.

On the target queue manager

- Local queue definition

  The target queue. The name of this queue must be the same as that specified in the remote queue name field of the remote queue definition on the source queue manager.
- Dead-letter queue definition

  Define a dead-letter queue to which undelivered messages can be written.

Related concepts:

"Creating a transmission queue"
Before a channel (other than a requester channel) can be started, the transmission queue must be defined as described in this section. The transmission queue must be named in the channel definition.

Creating a transmission queue:

Before a channel (other than a requester channel) can be started, the transmission queue must be defined as described in this section. The transmission queue must be named in the channel definition.

Define a local queue with the USAGE attribute set to XMITQ for each sending message channel. If you want to use a specific transmission queue in your remote queue definitions, create a remote queue as shown.

To create a transmission queue, use the WebSphere MQ Commands (MQSC), as shown in the following examples:

Create transmission queue example

```
DEFINE QLOCAL(QM2) DESCR('Transmission queue to QM2') USAGE(XMITQ)
```

Create remote queue example

```
DEFINE QREMOTE(PAYROLL) DESCR('Remote queue for QM2') +
XMITQ(QM2) RNAME(PAYROLL) RQMNAME(QM2)
```

Consider naming the transmission queue the queue manager name on the remote system, as shown in the examples.

Starting the channel:

When you put messages on the remote queue defined at the source queue manager, they are stored on the transmission queue until the channel is started. When the channel has been started, the messages are delivered to the target queue on the remote queue manager.

Start the channel on the sending queue manager using the START CHANNEL command. When you start the sending channel, the receiving channel is started automatically (by the listener) and the messages are sent to the target queue. Both ends of the message channel must be running for messages to be transferred.

Because the two ends of the channel are on different queue managers, they could have been defined with different attributes. To resolve any differences, there is an initial data negotiation between the two ends when the channel starts. In general, the two ends of the channel operate with the attributes needing the fewer resources. This enables larger systems to accommodate the lesser resources of smaller systems at the other end of the message channel.

The sending MCA splits large messages before sending them across the channel. They are reassembled at the remote queue manager. This is not apparent to the user.

An MCA can transfer messages using multiple threads. This process, called *pipelining* enables the MCA to transfer messages more efficiently, with fewer wait states. Pipelining improves channel performance.

Channel control function

The channel control function provides facilities for you to define, monitor, and control channels.

Commands are issued through panels, programs, or from a command line to the channel control function. The panel interface also displays channel status and channel definition data. You can use Programmable Command Formats or those WebSphere MQ commands (MQSC) and control commands that are detailed in "Monitoring and controlling channels on Windows, UNIX and Linux platforms" on page 807.

The commands fall into the following groups:

- Channel administration
- Channel control
- Channel status monitoring

Channel administration commands deal with the definitions of the channels. They enable you to:

- Create a channel definition
- Copy a channel definition
- Alter a channel definition
- Delete a channel definition

Channel control commands manage the operation of the channels. They enable you to:

- Start a channel
- Stop a channel
- Re-synchronize with partner (in some implementations)
- Reset message sequence numbers
- Resolve an in-doubt batch of messages
- Ping; send a test communication across the channel

Channel monitoring displays the state of channels, for example:

- Current channel settings
- Whether the channel is active or inactive
- Whether the channel terminated in a synchronized state

For more information about defining, controlling and monitoring channels, see the following subtopics:

Preparing channels:

Before trying to start a message channel or MQI channel, you must prepare the channel. You must make sure that all the attributes of the local and remote channel definitions are correct and compatible.

Channel attributes describes the channel definitions and attributes.

Although you set up explicit channel definitions, the channel negotiations carried out when a channel starts, might override one or other of the values defined. This behavior is normal, and not apparent to the user, and has been arranged in this way so that otherwise incompatible definitions can work together.

Auto-definition of receiver and server-connection channels

In WebSphere MQ on all platforms except z/OS, if there is no appropriate channel definition, then for a receiver or server-connection channel that has auto-definition enabled, a definition is created automatically. The definition is created using:

1. The appropriate model channel definition, SYSTEM.AUTO.RECEIVER, or SYSTEM.AUTO.SVRCONN. The model channel definitions for auto-definition are the same as the system defaults, SYSTEM.DEF.RECEIVER, and SYSTEM.DEF.SVRCONN, except for the description field, which is "Auto-defined by" followed by 49 blanks. The systems administrator can choose to change any part of the supplied model channel definitions.
2. Information from the partner system. The values from the partner are used for the channel name and the sequence number wrap value.
3. A channel exit program, which you can use to alter the values created by the auto-definition. See Channel auto-definition exit program.

The description is then checked to determine whether it has been altered by an auto-definition exit or because the model definition has been changed. If the first 44 characters are still "Auto-defined by" followed by 29 blanks, the queue manager name is added. If the final 20 characters are still all blanks the local time and date are added.

When the definition has been created and stored the channel start proceeds as though the definition had always existed. The batch size, transmission size, and message size are negotiated with the partner.

Defining other objects

Before a message channel can be started, both ends must be defined (or enabled for auto-definition) at their queue managers. The transmission queue it is to serve must be defined to the queue manager at the sending end. The communication link must be defined and available. It might be necessary for you to prepare other WebSphere MQ objects, such as remote queue definitions, queue manager alias definitions, and reply-to queue alias definitions, to implement the scenarios described in “Connecting applications using distributed queuing” on page 759.

For information about defining MQI channels, see “Defining MQI channels” on page 846.

Multiple message channels per transmission queue

It is possible to define more than one channel per transmission queue, but only one of these channels can be active at any one time. Consider this option for the provision of alternative routes between queue managers for traffic balancing and link failure corrective action. A transmission queue cannot be used by another channel if the previous channel to use it terminated leaving a batch of messages in-doubt at the sending end. For more information, see “In-doubt channels” on page 797.

Starting a channel

A channel can be caused to start transmitting messages in one of four ways. It can be:
- Started by an operator (not receiver, cluster-receiver, or server-connection channels).
- Triggered from the transmission queue. This method applies to sender channels and fully qualified server channels (those channels which specify a CONNAME) only. You must prepare the necessary objects for triggering channels.
- Started from an application program (not receiver, cluster-receiver, or server-connection channels).
- Started remotely from the network by a sender, cluster-sender, requester, server, or client-connection channel. Receiver, cluster-receiver, and possibly server and requester channel transmissions, are started this way; so are server-connection channels. The channels themselves must already be started (that is, enabled).

Note: Because a channel is 'started' it is not necessarily transmitting messages. Instead, it might be 'enabled' to start transmitting when one of the four events previously described occurs. The enabling and disabling of a channel is achieved using the START and STOP operator commands.

Channel states:

A channel can be in one of many states at any time. Some states also have substates. From a given state a channel can move into other states.

Figure 119 shows the hierarchy of all possible channel states and the substates that apply to each of the channel states.

Figure 120 on page 790 shows the links between channel states. These links apply to all types of message channel and server-connection channels.

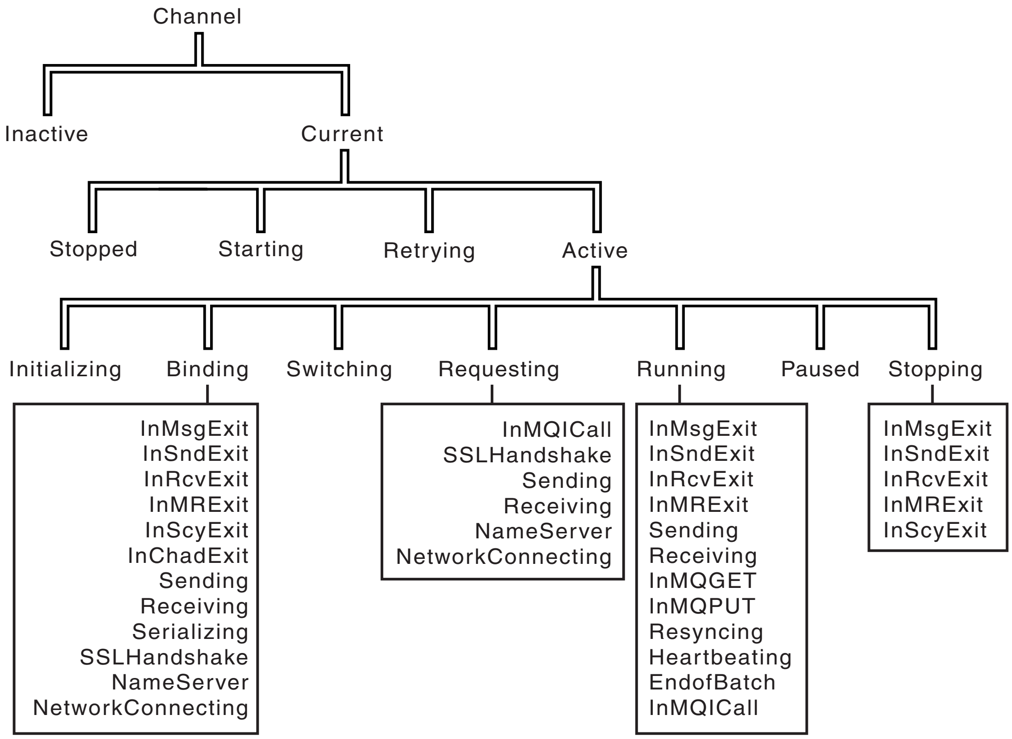


Figure 119. Channel states and substates

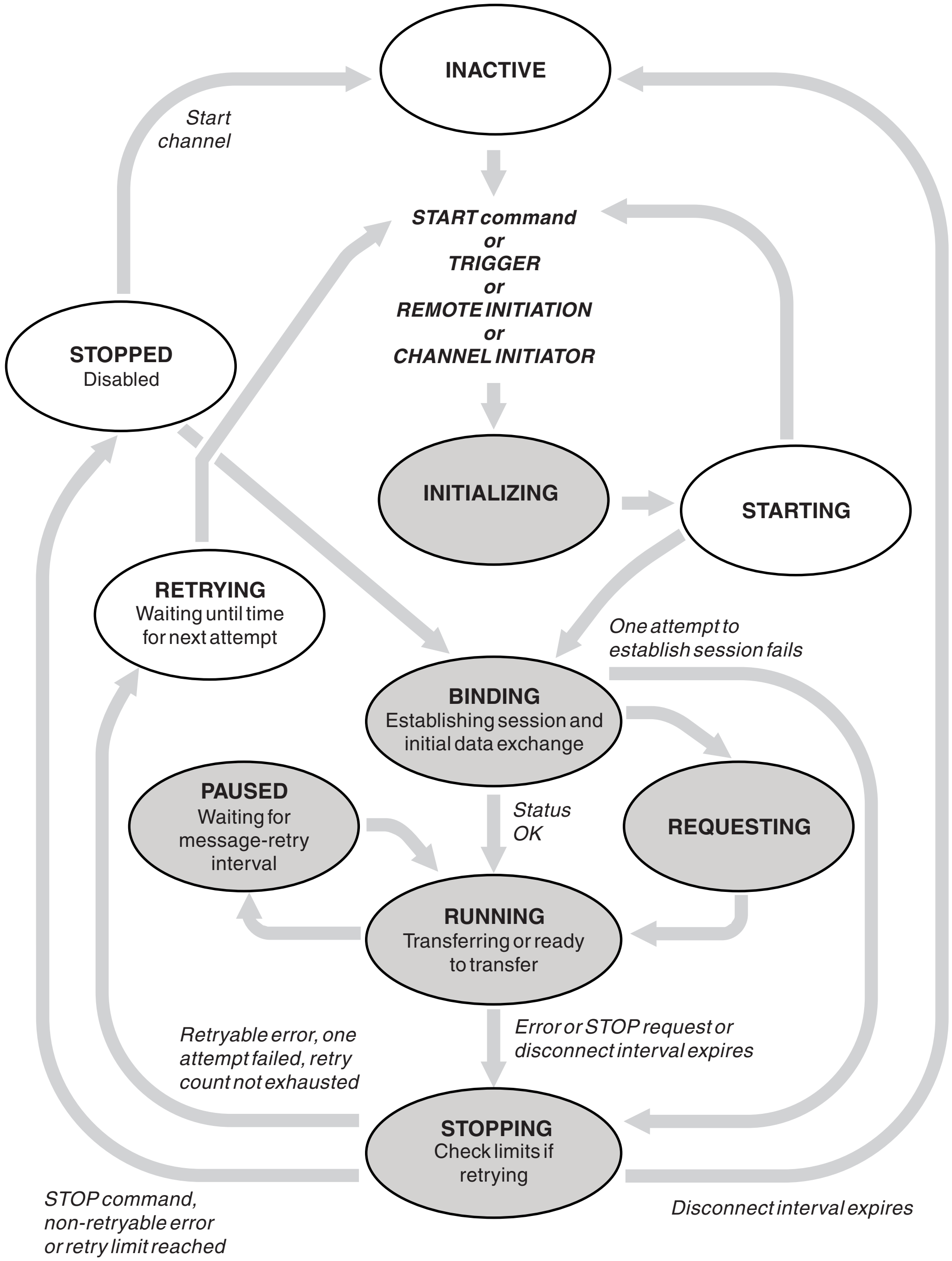

INACTIVE
Start channel
START command or TRIGGER or REMOTE INITIATION or CHANNEL INITIATOR
STOPPED
Disabled
INITIALIZING
STARTING
RETRYING
Waiting until time for next attempt
One attempt to establish session fails
BINDING
Establishing session and initial data exchange
PAUSED
Waiting for message-retry interval
Status OK
REQUESTING
RUNNING
Transferring or ready to transfer
Retryable error, one attempt failed, retry count not exhausted
Error or STOP request or disconnect interval expires
STOPPING
Check limits if retrying
STOP command, non-retryable error or retry limit reached
Disconnect interval expires

Figure 120. Flows between channel states

Current and active

A channel is *current* if it is in any state other than inactive. A current channel is *active* unless it is in RETRYING, STOPPED, or STARTING state. When a channel is active, it is consuming resource and a process or thread is running. The seven possible states of an active channel (INITIALIZING, BINDING, SWITCHING, REQUESTING, RUNNING, PAUSED, or STOPPING) are highlighted in Figure 120.

An active channel can also show a substate giving more detail of exactly what the channel is doing. The substates for each state are shown in Figure 119 on page 789.

Current and active:

The channel is “current” if it is in any state other than inactive. A current channel is “active” unless it is in RETRYING, STOPPED, or STARTING state.

If a channel is "active" it might also show a substate giving more detail of exactly what the channel is doing.

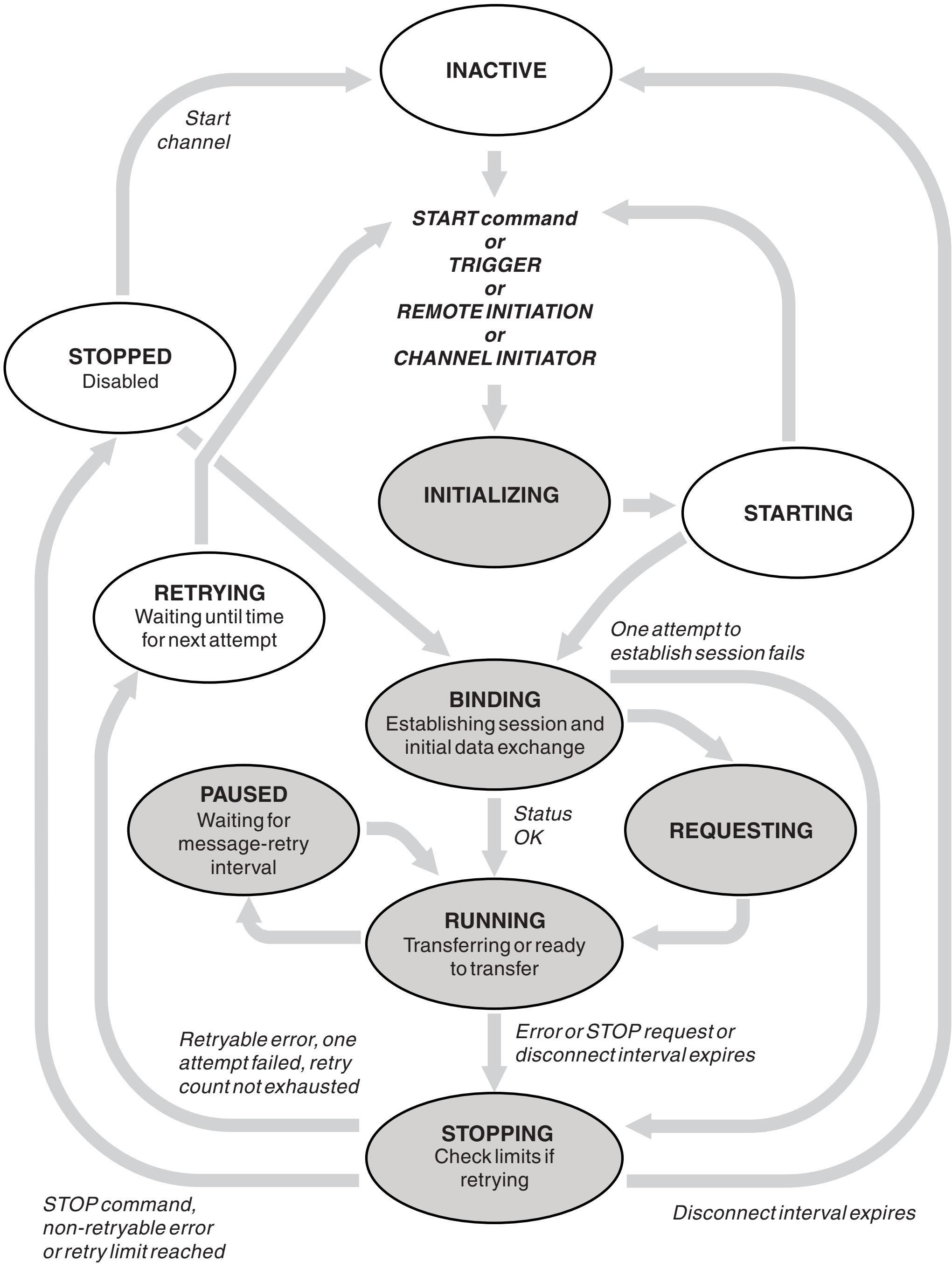

INACTIVE
Start channel
START command or TRIGGER or REMOTE INITIATION or CHANNEL INITIATOR
STOPPED
Disabled
INITIALIZING
STARTING
RETRYING
Waiting until time for next attempt
One attempt to establish session fails
BINDING
Establishing session and initial data exchange
PAUSED
Waiting for message-retry interval
Status OK
REQUESTING
RUNNING
Transferring or ready to transfer
Retryable error, one attempt failed, retry count not exhausted
Error or STOP request or disconnect interval expires
STOPPING
Check limits if retrying
STOP command, non-retryable error or retry limit reached
Disconnect interval expires

Figure 121. Flows between channel states

Note:

1. When a channel is in one of the six states highlighted in Figure 121 on page 791 (INITIALIZING, BINDING, REQUESTING, RUNNING, PAUSED, or STOPPING), it is consuming resource and a process or thread is running; the channel is *active*.
2. When a channel is in STOPPED state, the session might be active because the next state is not yet known.

Specifying the maximum number of current channels

You can specify the maximum number of channels that can be current at one time. This number is the number of channels that have entries in the channel status table, including channels that are retrying and channels that are stopped. Specify this using the queue manager configuration file for UNIX and Linux systems, or the WebSphere MQ Explorer. For more information about the values set using the initialization or the configuration file see Configuration file stanzas for distributed queuing. For more information about specifying the maximum number of channels, see Administering IBM WebSphere MQ for WebSphere MQ for UNIX and Linux systems, and Windows systems.

Note:

1. Server-connection channels are included in this number.
2. A channel must be current before it can become active. If a channel is started, but cannot become current, the start fails.

Specifying the maximum number of active channels

You can also specify the maximum number of active channels to prevent your system being overloaded by many starting channels. If you use this method, set the disconnect interval attribute to a low value to allow waiting channels to start as soon as other channels terminate.

Each time a channel that is retrying attempts to establish connection with its partner, it must become an active channel. If the attempt fails, it remains a current channel that is not active, until it is time for the next attempt. The number of times that a channel retries, and how often, is determined by the retry count and retry interval channel attributes. There are short and long values for both these attributes. See Channel attributes for more information.

When a channel has to become an active channel (because a START command has been issued, or because it has been triggered, or because it is time for another retry attempt), but is unable to do so because the number of active channels is already at the maximum value, the channel waits until one of the active slots is freed by another channel instance ceasing to be active. If, however, a channel is starting because it is being initiated remotely, and there are no active slots available for it at that time, the remote initiation is rejected.

Whenever a channel, other than a requester channel, is attempting to become active, it goes into the STARTING state. This state occurs even if there is an active slot immediately available, although it is only in the STARTING state for a short time. However, if the channel has to wait for an active slot, it is in STARTING state while it is waiting.

Requester channels do not go into STARTING state. If a requester channel cannot start because the number of active channels is already at the limit, the channel ends abnormally.

Whenever a channel, other than a requester channel, is unable to get an active slot, and so waits for one, a message is written to the log and an event is generated. When a slot is later freed and the channel is able to acquire it, another message and event are generated. Neither of these events and messages are generated if the channel is able to acquire a slot straight away.

If a STOP CHANNEL command is issued while the channel is waiting to become active, the channel goes to STOPPED state. A Channel-Stopped event is raised.

Server-connection channels are included in the maximum number of active channels.

For more information about specifying the maximum number of active channels, see Administering IBM WebSphere MQ for WebSphere MQ for UNIX and Linux systems, and Windows systems.

Channel errors:

Errors on channels cause the channel to stop further transmissions. If the channel is a sender or server, it goes to RETRY state because it is possible that the problem might clear itself. If it cannot go to RETRY state, the channel goes to STOPPED state.

For sending channels, the associated transmission queue is set to GET(DISABLED) and triggering is turned off. (A STOP command with STATUS(STOPPED) takes the side that issued it to STOPPED state; only expiry of the disconnect interval or a STOP command with STATUS(INACTIVE) makes it end normally and become inactive.) Channels that are in STOPPED state need operator intervention before they can restart (see "Restarting stopped channels" on page 797).

Note: For UNIX, Linux and Windows systems, a channel initiator must be running for retry to be attempted. If the channel initiator is not available, the channel becomes inactive and must be manually restarted. If you are using a script to start the channel, ensure that the channel initiator is running before you try to run the script.

Long retry count (LONGRTY) describes how retrying works. If the error clears, the channel restarts automatically, and the transmission queue is re-enabled. If the retry limit is reached without the error clearing, the channel goes to STOPPED state. A stopped channel must be restarted manually by the operator. If the error is still present, it does not retry again. When it does start successfully, the transmission queue is re-enabled.

If the queue manager stops while a channel is in RETRYING or STOPPED status, the channel status is remembered when the queue manager is restarted. However, the channel status for the SVRCONN channel type is reset if the queue manager stops while the channel is in STOPPED status.

If a channel is unable to put a message to the target queue because that queue is full or put inhibited, the channel can retry the operation a number of times (specified in the message-retry count attribute) at a time interval (specified in the message-retry interval attribute). Alternatively, you can write your own message-retry exit that determines which circumstances cause a retry, and the number of attempts made. The channel goes to PAUSED state while waiting for the message-retry interval to finish.

See Channel attributes for information about the channel attributes, and Channel-exit programs for messaging channels for information about the message-retry exit.

Server-connection channel limits:

You can set server-connection channel limits to prevent client applications from exhausting queue manager channel resources, **MAXINST**, and to prevent a single client application from exhausting server-connection channel capacity, **MAXINSTC**.

A maximum total number of channels can be active at any time on an individual queue manager. The total number of server-connection channel instances are included in the maximum number of active channels.

If you do not specify the maximum number of simultaneous instances of a server-connection channel that can be started, then it is possible for a single client application, connecting to a single server-connection channel, to exhaust the maximum number of active channels that are available. When the maximum number of active channels is reached, it prevents any other channels from being started on the queue manager. To avoid this situation, you must limit the number of simultaneous instances of an individual server-connection channel that can be started, regardless of which client started them.

If the value of the limit is reduced to below the currently running number of instances of the server connection channel, even to zero, then the running channels are not affected. New instances cannot be started until sufficient existing instances have ceased to run so that the number of currently running instances is less than the value of the limit.

Also, many different client-connection channels can connect to an individual server-connection channel. The limit on the number of simultaneous instances of an individual server-connection channel that can be started, regardless of which client started them, prevents any client from exhausting the maximum active channel capacity of the queue manager. If you do not also limit the number of simultaneous instances of an individual server-connection channel that can be started from an individual client, then it is possible for a single, faulty client application to open so many connections that it exhausts the channel capacity allocated to an individual server-connection channel, and therefore prevents other clients that need to use the channel from connecting to it. To avoid this situation, you must limit the number of simultaneous instances of an individual server-connection channel that can be started from an individual client.

If the value of the individual client limit is reduced below the number of instances of the server-connection channel that are currently running from individual clients, even to zero, then the running channels are not affected. However, new instances of the server-connection channel cannot be started from an individual client that exceeds the new limit until sufficient existing instances from that client have ceased to run so that the number of currently running instances is less than the value of this parameter.

Checking that the other end of the channel is still available:

You can use the heartbeat interval, the keep alive interval, and the receive timeout, to check that the other end of the channel is available.

Heartbeats

You can use the heartbeat interval channel attribute to specify that flows are to be passed from the sending MCA when there are no messages on the transmission queue, as is described in Heartbeat interval (HBINT).

Keep alive

In WebSphere MQ for UNIX, Linux, and Windows systems, if you are using TCP as your transport protocol, you can set `keepalive=yes`. If you specify this option, TCP periodically checks that the other end of the connection is still available. It is not, the channel is terminated. This option is described in Keepalive Interval (KAINT).

If you have unreliable channels that report TCP errors, use of the **`Keepalive`** option means that your channels are more likely to recover.

You can specify time intervals to control the behavior of the **`Keepalive`** option. When you change the time interval, only TCP/IP channels started after the change are affected. Ensure that the value that you choose for the time interval is less than the value of the disconnect interval for the channel.

For more information about using the **`Keepalive`** option, see the KAINT parameter in the DEFINE CHANNEL command.

Receive timeout

If you are using TCP as your transport protocol, the receiving end of an idle non-MQI channel connection is also closed if no data is received for a period. This period, the *receive time-out* value, is determined according to the HBINT (heartbeat interval) value.

In WebSphere MQ for UNIX, Linux, and Windows systems, the *receive time-out* value is set as follows:

1. For an initial number of flows, before any negotiation takes place, the *receive time-out* value is twice the HBINT value from the channel definition.
2. After the channels negotiate an HBINT value, if HBINT is set to less than 60 seconds, the *receive time-out* value is set to twice this value. If HBINT is set to 60 seconds or more, the *receive time-out* value is set to 60 seconds greater than the value of HBINT.

Note:

1. If either of the values is zero, there is no timeout.
2. For connections that do not support heartbeats, the HBINT value is negotiated to zero in step 2 and hence there is no timeout, so you must use TCP/IP KEEPALIVE.
3. For client connections that use sharing conversations, heartbeats can flow across the channel (from both ends) all the time, not just when an MQGET is outstanding.
4. For client connections where sharing conversations are not in use, heartbeats are flowed from the server only when the client issues an MQGET call with wait. Therefore, you are not recommended to set the heartbeat interval too small for client channels. For example, if the heartbeat is set to 10 seconds, an MQCMIT call fails (with MQRC_CONNECTION_BROKEN) if it takes longer than 20 seconds to commit because no data flowed during this time. This can happen with large units of work. However, it does not happen if appropriate values are chosen for the heartbeat interval because only MQGET with wait takes significant periods of time.

   Provided SHARECNV is not zero, the client uses a full duplex connection, which means that the client can (and does) heartbeat during all MQI calls
5. In WebSphere MQ Version 7 Client channels, heartbeats can flow from both the server as well as the client side. The timeout at either end is based upon 2*HBINT for HBINTs of less than 60 seconds and HBINT+60 for HBINTs of over 60 seconds.
6. Canceling the connection after twice the heartbeat interval is valid because a data or heartbeat flow is expected at least at every heartbeat interval. Setting the heartbeat interval too small, however, can cause problems, especially if you are using channel exits. For example, if the HBINT value is one second, and a send or receive exit is used, the receiving end waits for only 2 seconds before canceling the channel. If the MCA is performing a task such as encrypting the message, this value might be too short.

Adopting an MCA:

The Adopt MCA function enables WebSphere MQ to cancel a receiver channel and start a new one in its place.

If a channel suffers a communications failure, the receiver channel could be left in a 'communications receive' state. When communications are re-established the sender channel attempts to reconnect. If the remote queue manager finds that the receiver channel is already running it does not allow another version of the same receiver channel to be started. This problem requires user intervention to rectify the problem or the use of system keepalive.

The Adopt MCA function solves the problem automatically. It enables WebSphere MQ to cancel a receiver channel and to start a new one in its place.

The function can be set up with various options. For distributed platforms, see Administering.

Stopping and quiescing channels:

This topic explains how you can stop and quiesce a channel before the disconnect time interval expires.

Message channels are designed to be long-running connections between queue managers with orderly termination controlled only by the disconnect interval channel attribute. This mechanism works well unless the operator needs to terminate the channel before the disconnect time interval expires. This need can occur in the following situations:

- System quiesce
- Resource conservation
- Unilateral action at one end of a channel

In this case, you can stop the channel. You can do this using:

- the STOP CHANNEL MQSC command
- the Stop Channel PCF command
- the WebSphere MQ Explorer

There are three options for stopping channels using these commands:

QUIESCE
The QUIESCE option attempts to end the current batch of messages before stopping the channel.

FORCE
The FORCE option attempts to stop the channel immediately and might require the channel to resynchronize when it restarts because the channel might be left in doubt.

TERMINATE
The TERMINATE option attempts to stop the channel immediately, and terminates the thread or process of the channel.

All these options leave the channel in a STOPPED state, requiring operator intervention to restart it.

Stopping the channel at the sending end is effective but does require operator intervention to restart. At the receiving end of the channel, things are much more difficult because the MCA is waiting for data from the sending side, and there is no way to initiate an *orderly* termination of the channel from the receiving side; the stop command is pending until the MCA returns from its wait for data.

Consequently there are three recommended ways of using channels, depending upon the operational characteristics required:

- If you want your channels to be long running, note that there can be orderly termination only from the sending end. When channels are interrupted, that is, stopped, operator intervention (a START CHANNEL command) is required in order to restart them.
- If you want your channels to be active only when there are messages for them to transmit, set the disconnect interval to a fairly low value. The default setting is high and so is not recommended for channels where this level of control is required. Because it is difficult to interrupt the receiving channel, the most economical option is to have the channel automatically disconnect and reconnect as the workload demands. For most channels, the appropriate setting of the disconnect interval can be established heuristically.
- You can use the heartbeat-interval attribute to cause the sending MCA to send a heartbeat flow to the receiving MCA during periods in which it has no messages to send. This action releases the receiving MCA from its wait state and gives it an opportunity to quiesce the channel without waiting for the disconnect interval to expire. Give the heartbeat interval a lower value than the value of the disconnect interval.

  Note:

1. You are advised to set the disconnect interval to a low value, or to use heartbeats, for server channels. This low value is to allow for the case where the requester channel ends abnormally (for example, because the channel was canceled) when there are no messages for the server channel to send. If the disconnect interval is set high and heartbeats are not in use, the server does not detect that the requester has ended (which it will only do the next time it tries to send a message to the requester). While the server is still running, it holds the transmission queue open for exclusive input in order to get any more messages that arrive on the queue. If an attempt is made to restart the channel from the requester, the start request receives an error because the server still has the transmission queue open for exclusive input. It is necessary to stop the server channel, and then restart the channel from the requester again.

Restarting stopped channels:

When a channel goes into STOPPED state, you have to restart the channel manually.

To do restart the channel, issue one of the following commands:
- The START CHANNEL MQSC command
- The Start Channel PCF command
- the WebSphere MQ Explorer

For sender or server channels, when the channel entered the STOPPED state, the associated transmission queue was set to GET(DISABLED) and triggering was set off. When the start request is received, these attributes are reset automatically.

If the queue manager (on distributed platforms) stops while a channel is in RETRYING or STOPPED status, the channel status is remembered when the queue manager is restarted. However, the channel status for the SVRCONN channel type is reset if the queue manager stops while the channel is in STOPPED status.

In-doubt channels:

An in-doubt channel is a channel that is in doubt with a remote channel about which messages have been sent and received.

Note the distinction between this and a queue manager being in doubt about which messages should be committed to a queue.

You can reduce the opportunity for a channel to be placed in doubt by using the Batch Heartbeat channel parameter (BATCHHB). When a value for this parameter is specified, a sender channel checks that the remote channel is still active before taking any further action. If no response is received the receiver channel is considered to be no longer active. The messages can be rolled-back, and re-routed, and the sender-channel is not put in doubt. This reduces the time when the channel could be placed in doubt to the period between the sender channel verifying that the receiver channel is still active, and verifying that the receiver channel has received the sent messages. See Channel attributes for more information about the batch heartbeat parameter.

In-doubt channel problems are typically resolved automatically. Even when communication is lost, and a channel is placed in doubt with a message batch at the sender with receipt status unknown, the situation is resolved when communication is re-established. Sequence number and LUWID records are kept for this purpose. The channel is in doubt until LUWID information has been exchanged, and only one batch of messages can be in doubt for the channel.

You can, when necessary, resynchronize the channel manually. The term *manual* includes use of operators or programs that contain WebSphere MQ system management commands. The manual resynchronization process works as follows. This description uses MQSC commands, but you can also use the PCF equivalents.

1. Use the DISPLAY CHSTATUS command to find the last-committed logical unit of work ID (LUWID) for ***each*** side of the channel. Do this using the following commands:
   - For the in-doubt side of the channel:

     ```
     DISPLAY CHSTATUS(name) SAVED CURLUWID
     ```

     You can use the CONNAME and XMITQ parameters to further identify the channel.
   - For the receiving side of the channel:

     ```
     DISPLAY CHSTATUS(name) SAVED LSTLUWID
     ```

     You can use the CONNAME parameter to further identify the channel.

   The commands are different because only the sending side of the channel can be in doubt. The receiving side is never in doubt.

   On WebSphere MQ for IBM i, the DISPLAY CHSTATUS command can be executed from a file using the STRMQMMQSC command or the Work with MQM Channel Status CL command, WRKMQMCHST
2. If the two LUWIDs are the same, the receiving side has committed the unit of work that the sender considers to be in doubt. The sending side can now remove the in-doubt messages from the transmission queue and re-enable it. This is done with the following channel RESOLVE command:

   ```
   RESOLVE CHANNEL(name) ACTION(COMMIT)
   ```
3. If the two LUWIDs are different, the receiving side has not committed the unit of work that the sender considers to be in doubt. The sending side needs to retain the in-doubt messages on the transmission queue and re-send them. This is done with the following channel RESOLVE command:

   ```
   RESOLVE CHANNEL(name) ACTION(BACKOUT)
   ```

Once this process is complete the channel is no longer in doubt. The transmission queue can now be used by another channel, if required.

Problem determination:

There are two distinct aspects to problem determination - problems discovered when a command is being submitted, and problems discovered during operation of the channels.

Command validation

Commands and panel data must be free from errors before they are accepted for processing. Any errors found by the validation are immediately notified to the user by error messages.

Problem diagnosis begins with the interpretation of these error messages and taking corrective action.

Processing problems

Problems found during normal operation of the channels are notified to the system console or the system log. Problem diagnosis begins with the collection of all relevant information from the log, and continues with analysis to identify the problem.

Confirmation and error messages are returned to the terminal that initiated the commands, when possible.

WebSphere MQ produces accounting and statistical data, which you can use to identify trends in utilization and performance. On distributed platforms, this information is produced as PCF records, see Structure data types for details.

Messages and codes

For messages and codes to help with the primary diagnosis of the problem, see Diagnostic messages and reason codes.

Safety of messages

In addition to the typical recovery features of WebSphere MQ, distributed queue management ensures that messages are delivered properly by using a sync point procedure coordinated between the two ends of the message channel. If this procedure detects an error, it closes the channel so that you can investigate the problem, and keeps the messages safely in the transmission queue until the channel is restarted.

The sync point procedure has an added benefit in that it attempts to recover an *in-doubt* situation when the channel starts. (*In-doubt* is the status of a unit of recovery for which a sync point has been requested but the outcome of the request is not yet known.) Also associated with this facility are the two functions:

1. Resolve with commit or backout
2. Reset the sequence number

The use of these functions occurs only in exceptional circumstances because the channel recovers automatically in most cases.

Fast, nonpersistent messages

The nonpersistent message speed (NPMSPEED) channel attribute can be used to specify that any nonpersistent messages on the channel are to be delivered more quickly. For more information about this attribute, see Nonpersistent message speed (NPMSPEED).

If a channel terminates while fast, nonpersistent messages are in transit, the messages might be lost and it is up to the application to arrange for their recovery if required.

If the receiving channel cannot put the message to its destination queue then it is placed on the dead letter queue, if one has been defined. If not, the message is discarded.

Note: If the other end of the channel does not support the option, the channel runs at normal speed.

Undelivered Messages

For information about what happens when a message cannot be delivered, see “What happens when a message cannot be delivered?.”

What happens when a message cannot be delivered?

When a message cannot be delivered, the MCA can process it in several ways. It can try again, it can return-to-sender, or it can put it on the dead-letter queue.

Figure 122 on page 800 shows the processing that occurs when an MCA is unable to put a message to the destination queue. (The options shown do not apply on all platforms.)

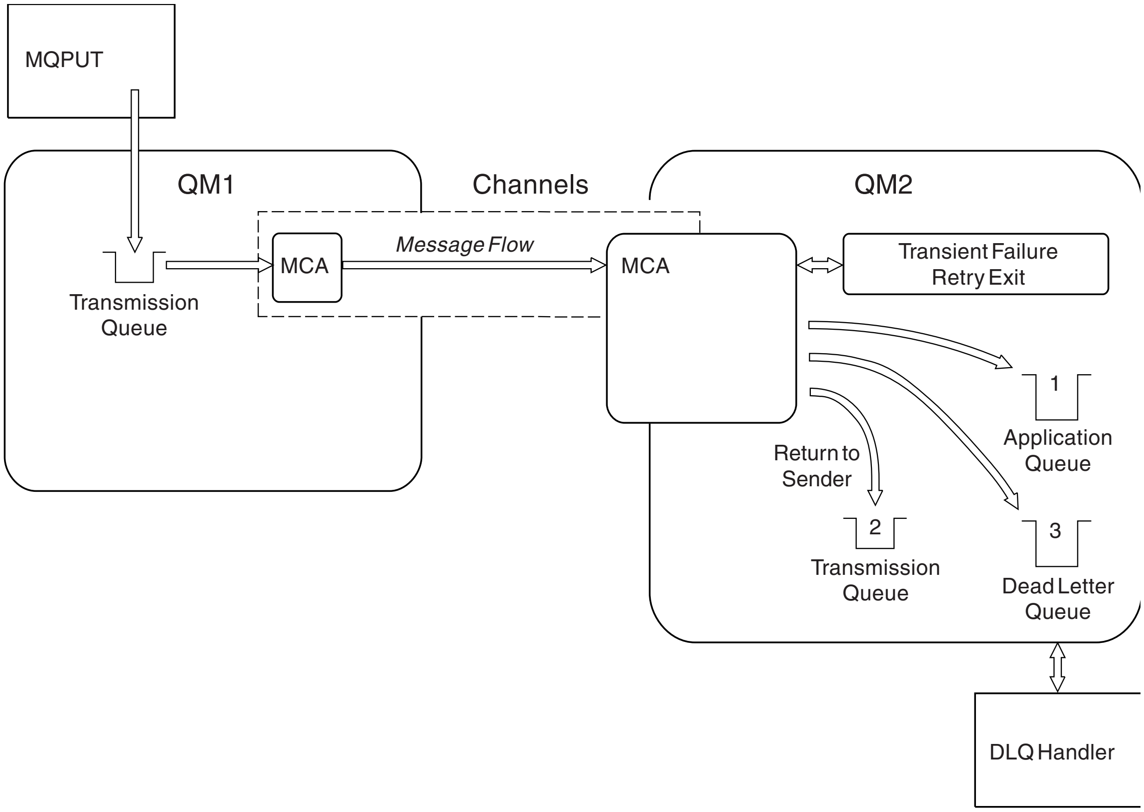


Figure 122. What happens when a message cannot be delivered

As shown in the figure, the MCA can do several things with a message that it cannot deliver. The action taken is determined by options specified when the channel is defined and on the MQPUT report options for the message.

1. Message-retry

   If the MCA is unable to put a message to the target queue for a reason that could be transitory (for example, because the queue is full), the MCA can wait and try the operation again later. You can determine if the MCA waits, for how long, and how many times it tries.

   - You can specify a message-retry time and interval for MQPUT errors when you define your channel. If the message cannot be put to the destination queue because the queue is full, or is inhibited for puts, the MCA tries the operation the number of times specified, at the time interval specified.
   - You can write your own message-retry exit. The exit enables you to specify under what conditions you want the MCA to try the MQPUT or MQOPEN operation again. Specify the name of the exit when you define the channel.

2. Return-to-sender

   If message-retry was unsuccessful, or a different type of error was encountered, the MCA can send the message back to the originator. To enable return-to-sender, you need to specify the following options in the message descriptor when you put the message to the original queue:

   - The MQRO_EXCEPTION_WITH_FULL_DATA report option
   - The MQRO_DISCARD_MSG report option
   - The name of the reply-to queue and reply-to queue manager

   If the MCA is unable to put the message to the destination queue, it generates an exception report containing the original message, and puts it on a transmission queue to be sent to the reply-to queue specified in the original message. (If the reply-to queue is on the same queue manager as the MCA, the message is put directly to that queue, not to a transmission queue.)

3. Dead-letter queue

   If a message cannot be delivered or returned, it is put on to the dead-letter queue (DLQ). You can use the DLQ handler to process the message. This processing is described in Handling undelivered messages with the WebSphere MQ dead-letter queue handler for IBM WebSphere MQ for UNIX, Linux. If the dead-letter queue is not available, the sending MCA leaves the message on the transmission queue, and the channel stops. On a fast channel, nonpersistent messages that cannot be written to a dead-letter queue are lost.

   On IBM WebSphere MQ Version 7.0, if no local dead-letter queue is defined, the remote queue is not available or defined, and there is no remote dead-letter queue, then the sender channel goes into RETRY and messages are automatically rolled back to the transmission queue.

Related information:

Use Dead-Letter Queue (USEDLQ)

Triggering channels

WebSphere MQ provides a facility for starting an application automatically when certain conditions on a queue are met. This facility is called triggering.

This explanation is intended as an overview of triggering concepts. For a complete description, see Starting WebSphere MQ applications using triggers.

For platform-specific information see the following:

- For Windows, see UNIX and Linux systems, “Triggering channels on UNIX, Linux and Windows systems.” on page 802

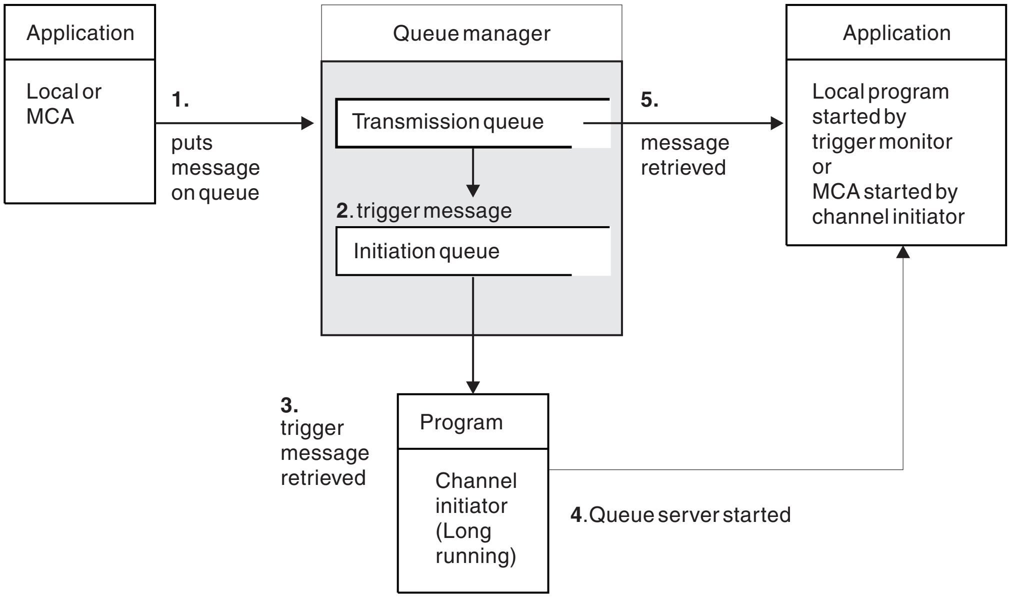


Figure 123. The concepts of triggering

The objects required for triggering are shown in Figure 123. It shows the following sequence of events:

1. The local queue manager places a message from an application or from a message channel agent (MCA) on the transmission queue.

2. When the triggering conditions are fulfilled, the local queue manager places a trigger message on the initiation queue.
3. The long-running channel initiator program monitors the initiation queue, and retrieves messages as they arrive.
4. The channel initiator processes the trigger messages according to information contained in them. This information might include the channel name, in which case the corresponding MCA is started.
5. The local application or the MCA, having been triggered, retrieves the messages from the transmission queue.

To set up this scenario, you need to:

- Create the transmission queue with the name of the initiation queue (that is, SYSTEM.CHANNEL.INITQ) in the corresponding attribute.
- Ensure that the initiation queue (SYSTEM.CHANNEL.INITQ) exists.
- Ensure that the channel initiator program is available and running. The channel initiator program must be provided with the name of the initiation queue in its start command.
- Optionally, create the process definition for the triggering, if it does not exist, and ensure that the *UserData* field contains the name of the channel it serves. Instead of creating a process definition, you can specify the channel name in the *TriggerData* attribute of the transmission queue. WebSphere MQ for UNIX, Linux and Windows systems, allow the channel name to be specified as blank, in which case the first available channel definition with this transmission queue is used.
- Ensure that the transmission queue definition contains the name of the process definition to serve it, (if applicable), the initiation queue name, and the triggering characteristics you feel are most suitable. The trigger control attribute allows triggering to be enabled, or not, as necessary.

Note:

1. The channel initiator program acts as a 'trigger monitor' monitoring the initiation queue used to start channels.
2. An initiation queue and trigger process can be used to trigger any number of channels.
3. Any number of initiation queues and trigger processes can be defined.
4. A trigger type of FIRST is recommended, to avoid flooding the system with channel starts.

Triggering channels on UNIX, Linux and Windows systems.

You can create a process definition in WebSphere MQ, defining processes to be triggered. Use the MQSC command DEFINE PROCESS to create a process definition naming the process to be triggered when messages arrive on a transmission queue. The USERDATA attribute of the process definition contains the name of the channel being served by the transmission queue.

Define the local queue (QM4), specifying that trigger messages are to be written to the initiation queue (IQ) to trigger the application that starts channel (QM3.TO.QM4):

```
DEFINE QLOCAL(QM4) TRIGGER INITQ(SYSTEM.CHANNEL.INITQ) PROCESS(P1) USAGE(XMITQ)
```

Define the application (process P1) to be started:

```
DEFINE PROCESS(P1) USERDATA(QM3.TO.QM4)
```

Alternatively, for WebSphere MQ for UNIX, Linux and Windows systems, you can eliminate the need for a process definition by specifying the channel name in the TRIGDATA attribute of the transmission queue.

Define the local queue (QM4). Specify that trigger messages are to be written to the default initiation queue SYSTEM.CHANNEL.INITQ, to trigger the application (process P1) that starts channel (QM3.TO.QM4):

```
DEFINE QLOCAL(QM4) TRIGGER INITQ(SYSTEM.CHANNEL.INITQ)
USAGE(XMITQ) TRIGDATA(QM3.TO.QM4)
```

If you do not specify a channel name, the channel initiator searches the channel definition files until it finds a channel that is associated with the named transmission queue.

Related concepts:

"Starting and stopping the channel initiator"
Triggering is implemented using the channel initiator process.

"Connecting applications using distributed queuing" on page 759
This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

Related information:

Channel programs on UNIX, Linux, and Windows systems

Starting and stopping the channel initiator:

Triggering is implemented using the channel initiator process.

This channel initiator process is started with the MQSC command START CHINIT. Unless you are using the default initiation queue, specify the name of the initiation queue on the command. For example, to use the START CHINIT command to start queue IQ for the default queue manager, enter:

```
START CHINIT INITQ(IQ)
```

By default, a channel initiator is started automatically using the default initiation queue, SYSTEM.CHANNEL.INITQ. If you want to start all your channel initiators manually, follow these steps:

1. Create and start the queue manager.
2. Alter the queue manager's SCHINIT property to MANUAL
3. End and restart the queue manager

In Linux and Windows systems, a channel initiator is started automatically. The number of channel initiators that you can start is limited. The default and maximum value is 3. You can change this using MAXINITIATORS in the qm.ini file for UNIX and Linux systems, and in the registry for Windows systems.

See WebSphere MQ Control commands for details of the run channel initiator command **runmqchi**, and the other control commands.

Stopping the channel initiator

The default channel initiator is started automatically when you start a queue manager. All channel initiators are stopped automatically when a queue manager is stopped.

Initialization and configuration files

The handling of channel initialization data depends on your WebSphere MQ platform.

Windows, UNIX and Linux systems

In WebSphere MQ for Windows, UNIX and Linux systems, there are *configuration files* to hold basic configuration information about the WebSphere MQ installation.

There are two configuration files: one applies to the machine, the other applies to an individual queue manager.

WebSphere MQ configuration file

This file holds information relevant to all the queue managers on the WebSphere MQ system. The file is called mqs.ini. It is fully described in the Administering for WebSphere MQ for Windows, UNIX and Linux systems.

Queue manager configuration file

This file holds configuration information relating to one particular queue manager. The file is called qm.ini.

It is created during queue manager creation and can hold configuration information relevant to any aspect of the queue manager. Information held in the file includes details of how the configuration of the log differs from the default in WebSphere MQ configuration file.

The queue manager configuration file is held in the root of the directory tree occupied by the queue manager. For example, for the DefaultPath attributes, the queue manager configuration files for a queue manager called QMNAME would be:

For UNIX and Linux systems:

```
/var/mqm/qmgrs/QMNAME/qm.ini
```

An excerpt of a qm.ini file follows. It specifies that the TCP/IP listener is to listen on port 2500, the maximum number of current channels is to be 200, and the maximum number of active channels is to be 100.

```
TCP:
  Port=2500
CHANNELS:
  MaxChannels=200
  MaxActiveChannels=100
```

You can specify a range of TCP/IP ports to be used by an outbound channel. One method is to use the qm.ini file to specify the start and end of a range of port values. The following example shows a qm.ini file specifying a range of channels:

```
TCP:
  StrPort=2500
  EndPort=3000
CHANNELS:
  MaxChannels=200
  MaxActiveChannels=100
```

If you specify a value for StrPort or EndPort then you must specify a value for both. The value of EndPort must always be greater than the value of StrPort.

The channel tries to use each of the port values in the range specified. When the connection is successful, the port value is the port that the channel then uses.

For Windows systems:

```
C:\Program Files\IBM\WebSphere MQ\qmgrs\QMNAME\qm.ini
```

For more information about qm.ini files, see Configuration file stanzas for distributed queuing.

Data conversion

WebSphere MQ messages might require data conversion when sent between queues on different queue managers.

A WebSphere MQ message consists of two parts:

- Control information in a message descriptor
- Application data

Either of the two parts might require data conversion when sent between queues on different queue managers. For information about application data conversion, see Application data conversion.

Writing your own message channel agents

WebSphere MQ allows you to write your own message channel agent (MCA) programs or to install one from an independent software vendor.

You might want to write your own MCA programs to make WebSphere MQ interoperate over your own proprietary communications protocol, or to send messages over a protocol that WebSphere MQ does not support. (You cannot write your own MCA to interoperate with a WebSphere MQ-supplied MCA at the other end.)

If you decide to use an MCA that was not supplied by WebSphere MQ, you must consider the following points.

Message sending and receiving

You must write a sending application that gets messages from wherever your application puts them, for example from a transmission queue, and sends them out on a protocol with which you want to communicate. You must also write a receiving application that takes messages from this protocol and puts them onto destination queues. The sending and receiving applications use the message queue interface (MQI) calls, not any special interfaces.

You must ensure that messages are only delivered once. Sync point coordination can be used to help with this delivery.

Channel control function

You must provide your own administration functions to control channels. You cannot use WebSphere MQ channel administration functions either for configuring (for example, the DEFINE CHANNEL command) or monitoring (for example, DISPLAY CHSTATUS) your channels.

Initialization file

You must provide your own initialization file, if you require one.

Application data conversion

You probably want to allow for data conversion for messages you send to a different system. If so, use the MQGMO_CONVERT option on the MQGET call when retrieving messages from wherever your application puts them, for example the transmission queue.

User exits

Consider whether you need user exits. If so, you can use the same interface definitions that WebSphere MQ uses.

Triggering

If your application puts messages to a transmission queue, you can set up the transmission queue attributes so that your sending MCA is triggered when messages arrive on the queue.

Channel initiator

You might must provide your own channel initiator.

Other things to consider for distributed queue management

Other topics to consider when preparing WebSphere MQ for distributed queue management. This topic covers Undelivered-message queue, Queues in use, System extensions and user-exit programs, and Running channels and listeners as trusted applications.

Undelivered-message queue

To ensure that messages arriving on the undelivered-message queue (also known as the dead-letter queue or DLQ) are processed, create a program that can be triggered or run at regular intervals to handle these messages. A DLQ handler is provided with WebSphere MQ on UNIX and Linux systems; for more information, see The sample DLQ handler, amqsdlq.

Queues in use

MCAs for receiver channels can keep the destination queues open even when messages are not being transmitted. This results in the queues appearing to be "in use".

Maximum number of channels

See Configuration file stanzas for distributed queuing.

System extensions and user-exit programs

A facility is provided in the channel definition to enable extra programs to be run at defined times during the processing of messages. These programs are not supplied with WebSphere MQ, but can be provided by each installation according to local requirements.

In order to run, these user-exit programs must have predefined names and be available on call to the channel programs. The names of the user-exit programs are included in the message channel definitions.

There is a defined control block interface for handing over control to these programs, and for handling the return of control from these programs.

The precise places where these programs are called, and details of control blocks and names, are to be found in Channel-exit programs for messaging channels.

Running channels and listeners as trusted applications

If performance is an important consideration in your environment and your environment is stable, you can run your channels and listeners as trusted, using the FASTPATH binding. There are two factors that influence whether channels and listeners run as trusted:

- The environment variable MQ_CONNECT_TYPE=FASTPATH or MQ_CONNECT_TYPE=STANDARD. This is case-sensitive. If you specify a value that is not valid it is ignored.
- MQIBindType in the Channels stanza of the `qm.ini` or registry file. You can set this to FASTPATH or STANDARD and it is not case-sensitive. The default is STANDARD.

You can use MQIBindType in association with the environment variable to achieve the required effect as follows:

MQIBindType	Environment variable	Result
STANDARD	UNDEFINED	STANDARD
FASTPATH	UNDEFINED	FASTPATH
STANDARD	STANDARD	STANDARD
FASTPATH	STANDARD	STANDARD
STANDARD	FASTPATH	STANDARD
FASTPATH	FASTPATH	FASTPATH
STANDARD	CLIENT	CLIENT
FASTPATH	CLIENT	STANDARD
STANDARD	LOCAL	STANDARD
FASTPATH	LOCAL	STANDARD

In summary, there are only two ways of actually making channels and listeners run as trusted:

1. By specifying MQIBindType=FASTPATH in `qm.ini` or registry and not specifying the environment variable.
2. By specifying MQIBindType=FASTPATH in `qm.ini` or registry and setting the environment variable to FASTPATH.

Consider running listeners as trusted, because listeners are stable processes. Consider running channels as trusted, unless you are using unstable channel exits or the command STOP CHANNEL MODE(TERMINATE).

Monitoring and controlling channels on Windows, UNIX and Linux platforms

For DQM you need to create, monitor, and control the channels to remote queue managers. You can control channels using commands, programs, WebSphere MQ Explorer, files for the channel definitions, and a storage area for synchronization information.

You can use the following types of command:

The WebSphere MQ commands (MQSC)

You can use the MQSC as single commands in an MQSC session in Windows, UNIX and Linux systems. To issue more complicated, or multiple, commands the MQSC can be built into a file that you then run from the command line. For details, see MQSC commands. This section gives some simple examples of using MQSC for distributed queuing.

The channel commands are a subset of the WebSphere MQ Commands (MQSC). You use MQSC and the control commands to:

- Create, copy, display, change, and delete channel definitions
- Start and stop channels, ping, reset channel sequence numbers, and resolve in-doubt messages when links cannot be re-established
- Display status information about channels

Control commands

You can also issue *control commands* at the command line for some of these functions. For details, see control commands.

Programmable command format commands

For details, see PCF commands.

WebSphere MQ Explorer

On UNIX, Linux and Windows systems, you can use the WebSphere MQ Explorer. This provides a graphical administration interface to perform administrative tasks as an alternative to using control commands or MQSC commands. Channel definitions are held as queue manager objects.

Each queue manager has a DQM component for controlling interconnections to compatible remote queue managers. A storage area holds sequence numbers and *logical unit of work (LUW)* identifiers. These are used for channel synchronization purposes.

For a list of the functions available to you when setting up and controlling message channels, using the different types of command, see Table 110 on page 809.

Related concepts:

"Getting started with objects" on page 810
Channels must be defined, and their associated objects must exist and be available for use, before a channel can be started. This section shows you how.

"Setting up communication for Windows" on page 817
When a distributed-queuing management channel is started, it tries to use the connection specified in the channel definition. For this to succeed, it is necessary for the connection to be defined and available. This section explains how to do this using one of the four forms of communication for WebSphere MQ for Windows systems.

"Setting up communication on UNIX and Linux systems" on page 826
DQM is a remote queuing facility for WebSphere MQ. It provides channel control programs for the queue manager which form the interface to communication links, controllable by the system operator. The channel definitions held by distributed-queuing management use these connections.

Related information:

Channel programs on UNIX, Linux, and Windows systems

Message channel planning example for distributed platforms

Example configuration information

Channel attributes

Functions required for setting up and controlling channels

A number of WebSphere MQ functions might be needed to set up and control channels. The channel functions are explained in this topic.

You can create a channel definition using the default values supplied by WebSphere MQ, specifying the name of the channel, the type of channel you are creating, the communication method to be used, the transmission queue name and the connection name.

The channel name must be the same at both ends of the channel, and unique within the network. However, you must restrict the characters used to those that are valid for WebSphere MQ object names.

For other channel related functions, see the following topics:

- "Getting started with objects" on page 810
- "Creating associated objects" on page 811
- "Creating default objects" on page 811
- "Creating a channel" on page 811
- "Displaying a channel" on page 812
- "Displaying channel status" on page 812
- "Checking links using Ping" on page 813
- "Starting a channel" on page 813
- "Stopping a channel" on page 815
- "Renaming a channel" on page 815
- "Resetting a channel" on page 816

- “Resolving in-doubt messages on a channel” on page 816

Table 110 shows the full list of WebSphere MQ functions that you might need.

Table 110. Functions required in Windows, UNIX and Linux systems

Function	Control commands	MQSC	WebSphere MQ Explorer equivalent?
Queue manager functions			
Change queue manager		ALTER QMGR	Yes
Create queue manager	crtmqm		Yes
Delete queue manager	dltmqm		Yes
Display queue manager		DISPLAY QMGR	Yes
End queue manager	endmqm		Yes
Ping queue manager		PING QMGR	No
Start queue manager	strmqm		Yes
Command server functions			
Display command server	dspmqcsv		No
End command server	endmqcsv		No
Start command server	strmqcsv		No
Queue functions			
Change queue		ALTER QALIAS ALTER QLOCAL ALTER QMODEL ALTER QREMOTE See, ALTER queues .	Yes
Clear queue		CLEAR QLOCAL	Yes
Create queue		DEFINE QALIAS DEFINE QLOCAL DEFINE QMODEL DEFINE QREMOTE See, DEFINE queues .	Yes
Delete queue		DELETE QALIAS DELETE QLOCAL DELETE QMODEL DELETE QREMOTE See, DELETE queues .	Yes
Display queue		DISPLAY QUEUE	Yes
Process functions			
Change process		ALTER PROCESS	Yes
Create process		DEFINE PROCESS	Yes
Delete process		DELETE PROCESS	Yes
Display process		DISPLAY PROCESS	Yes
Channel functions			
Change channel		ALTER CHANNEL	Yes
Create channel		DEFINE CHANNEL	Yes

Table 110. Functions required in Windows, UNIX and Linux systems (continued)

Function	Control commands	MQSC	WebSphere MQ Explorer equivalent?
Delete channel		DELETE CHANNEL	Yes
Display channel		DISPLAY CHANNEL	Yes
Display channel status		DISPLAY CHSTATUS	Yes
End channel		STOP CHANNEL	Yes
Ping channel		PING CHANNEL	Yes
Reset channel		RESET CHANNEL	Yes
Resolve channel		RESOLVE CHANNEL	Yes
Run channel	runmqchl	START CHANNEL	Yes
Run channel initiator	runmqchi	START CHINIT	No
Run listener[1]	runmqlsr	START LISTENER	No
End listener	endmqlsr (Windows systems, AIX, HP-UX, and Solaris only)		No

Note:

1. A listener might be started automatically when the queue manager starts.

Getting started with objects

Channels must be defined, and their associated objects must exist and be available for use, before a channel can be started. This section shows you how.

Use the WebSphere MQ commands (MQSC) or the WebSphere MQ Explorer to:

1. Define message channels and associated objects
2. Monitor and control message channels

The associated objects you might need to define are:

- Transmission queues
- Remote queue definitions
- Queue manager alias definitions
- Reply-to queue alias definitions
- Reply-to local queues
- Processes for triggering (MCAs)
- Message channel definitions

The particular communication link for each channel must be defined and available before a channel can be run. For a description of how LU 6.2, TCP/IP, NetBIOS, SPX, and DECnet links are defined, see the particular communication guide for your installation. See also Example configuration information.

For more information about creating and working with objects, see the following subtopics:

Creating associated objects:

MQSC is used to create associated objects.

Use MQSC to create the queue and alias objects: transmission queues, remote queue definitions, queue manager alias definitions, reply-to queue alias definitions, and reply-to local queues.

Also create the definitions of processes for triggering (MCAs) in a similar way.

For an example showing how to create all the required objects see Message channel planning example for distributed platforms.

Creating default objects:

Default objects are created automatically when a queue manager is created. These objects are queues, channels, a process definition, and administration queues. After the default objects have been created, you can replace them at any time by running the strmqm command with the -c option.

When you use the crtmqm command to create a queue manager, the command also initiates a program to create a set of default objects.

1. Each default object is created in turn. The program keeps a count of how many objects are successfully defined, how many existed and were replaced, and how many unsuccessful attempts there were.
2. The program displays the results to you and if any errors occurred, directs you to the appropriate error log for details.

When the program has finished running, you can use the strmqm command to start the queue manager.

See The control commands for more information about the crtmqm and strmqm commands.

Changing the default objects

When you specify the -c option, the queue manager is started temporarily while the objects are created and is then shut down again. Issuing strmqm with the -c option refreshes existing system objects with the default values (for example, the MCAUSER attribute of a channel definition is set to blanks). You must use the strmqm command again, without the -c option, if you want to start the queue manager.

If you want to change the default objects, you can create your own version of the old amqscoma.tst file and edit it.

Creating a channel:

Create ***two*** channel definitions, one at each end of the connection. You create the first channel definition at the first queue manager. Then you create the second channel definition at the second queue manager, on the other end of the link.

Both ends must be defined using the ***same*** channel name. The two ends must have **compatible** channel types, for example: Sender and Receiver.

To create a channel definition for one end of the link use the MQSC command DEFINE CHANNEL. Include the name of the channel, the channel type for this end of the connection, a connection name, a description (if required), the name of the transmission queue (if required), and the transmission protocol. Also include any other attributes that you want to be different from the system default values for the required channel type, using the information you have gathered previously.

You are provided with help in deciding on the values of the channel attributes in Channel attributes.

Note: You are recommended to name all the channels in your network uniquely. Including the source and target queue manager names in the channel name is a good way to do this.

Create channel example

```
DEFINE CHANNEL(QM1.TO.QM2) CHLTYPE(SDR) +
DESCR('Sender channel to QM2') +
CONNAME(QM2) TRPTYPE(TCP) XMITQ(QM2) CONVERT(YES)
```

In all the examples of MQSC the command is shown as it appears in a file of commands, and as it is typed in Windows or UNIX or Linux systems. The two methods look identical, except that to issue a command interactively, you must first start an MQSC session. Type `runmqsc`, for the default queue manager, or `runmqsc` *`qmname`* where *`qmname`* is the name of the required queue manager. Then type any number of commands, as shown in the examples.

For portability, restrict the line length of your commands to 72 characters. Use the concatenation character, **+**, as shown to continue over more than one line. On Windows use Ctrl-z to end the entry at the command line. On UNIX and Linux systems, use Ctrl-d. Alternatively, on UNIX, Linux or Windows systems, use the **end** command.

Displaying a channel:

Use the MQSC command DISPLAY CHANNEL to display the attributes of a channel.

The ALL parameter of the DISPLAY CHANNEL command is assumed by default if no specific attributes are requested and the channel name specified is not generic.

The attributes are described in Channel attributes.

Display channel examples

```
DISPLAY CHANNEL(QM1.TO.QM2) TRPTYPE,CONVERT

DISPLAY CHANNEL(QM1.TO.*) TRPTYPE,CONVERT

DISPLAY CHANNEL(*) TRPTYPE,CONVERT

DISPLAY CHANNEL(QM1.TO.QMR34) ALL
```

Displaying channel status:

Use the MQSC command DISPLAY CHSTATUS, specifying the channel name and whether you want the current status of channels or the status of saved information.

DISPLAY CHSTATUS applies to all message channels. It does not apply to MQI channels other than server-connection channels.

Information displayed includes:

- Channel name
- Communication connection name
- In-doubt status of channel (where appropriate)
- Last sequence number
- Transmission queue name (where appropriate)
- The in-doubt identifier (where appropriate)
- The last committed sequence number
- Logical unit of work identifier
- Process ID

- Thread ID (Windows only)

Display channel status examples

```
DISPLAY CHSTATUS(*) CURRENT
```

```
DISPLAY CHSTATUS(QM1.TO.*) SAVED
```

The saved status does not apply until at least one batch of messages has been transmitted on the channel. Status is also saved when a channel is stopped (using the STOP CHL command) and when the queue manager is ended.

Checking links using Ping:

Use the MQSC command PING CHANNEL to exchange a fixed data message with the remote end.

Ping gives some confidence to the system supervisor that the link is available and functioning.

Ping does not involve the use of transmission queues and target queues. It uses channel definitions, the related communication link, and the network setup. It can only be used if the channel is not currently active.

It is available from sender and server channels only. The corresponding channel is started at the far side of the link, and performs the startup parameter negotiation. Errors are notified normally.

The result of the message exchange is presented as `Ping complete` or an error message.

Ping with LU 6.2

When Ping is invoked, by default no user ID or password flows to the receiving end. If user ID and password are required, they can be created at the initiating end in the channel definition. If a password is entered into the channel definition, it is encrypted by WebSphere MQ before being saved. It is then decrypted before flowing across the conversation.

Starting a channel:

Use the MQSC command START CHANNEL for sender, server, and requester channels. For applications to be able to exchange messages, you must start a listener program for inbound connections.

START CHANNEL is not necessary where a channel has been set up with queue manager triggering.

When started, the sending MCA reads the channel definitions and opens the transmission queue. A channel start-up sequence is issued, which remotely starts the corresponding MCA of the receiver or server channel. When they have been started, the sender and server processes await messages arriving on the transmission queue and transmit them as they arrive.

When you use triggering or run channels as threads, ensure that the channel initiator is available to monitor the initiation queue. The channel initiator is started by default as part of the queue manager.

However, TCP and LU 6.2 do provide other capabilities:

- For TCP on UNIX and Linux systems, inetd can be configured to start a channel. inetd is started as a separate process.
- For LU 6.2 in UNIX and Linux systems, configure your SNA product to start the LU 6.2 responder process.
- For LU 6.2 in Windows systems, using SNA Server you can use TpStart (a utility provided with SNA Server) to start a channel. TpStart is started as a separate process.

Use of the Start option always causes the channel to resynchronize, where necessary.

For the start to succeed:

- Channel definitions, local and remote, must exist. If there is no appropriate channel definition for a receiver or server-connection channel, a default one is created automatically if the channel is auto-defined. See Channel auto-definition exit program.
- Transmission queue must exist, and have no other channels using it.
- MCAs, local and remote, must exist.
- Communication link must be available.
- Queue managers must be running, local and remote.
- Message channel must not be already running.

A message is returned to the screen confirming that the request to start a channel has been accepted. For confirmation that the start command has succeeded, check the error log, or use DISPLAY CHSTATUS. The error logs are:

Windows

MQ_INSTALLATION_PATH\qmgrs\qmname\errors\AMQERR01.LOG (for each queue manager called qmname)

MQ_INSTALLATION_PATH\qmgrs\@SYSTEM\errors\AMQERR01.LOG (for general errors)

MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.

Note: On Windows systems, you still also get a message in the Windows systems application event log.

UNIX and Linux systems

/var/mqm/qmgrs/qmname/errors/AMQERR01.LOG (for each queue manager called qmname)

/var/mqm/qmgrs/@SYSTEM/errors/AMQERR01.LOG (for general errors)

On Windows, UNIX and Linux systems, use the runmqlsr command to start the WebSphere MQ listener process. By default, any inbound requests for channel attachment causes the listener process to start MCAs as threads of the amqrmppa process.

```
runmqlsr -t tcp -m QM2
```

For outbound connections, you must start the channel in one of the following three ways:

1. Use the MQSC command START CHANNEL, specifying the channel name, to start the channel as a process or a thread, depending on the MCATYPE parameter. (If channels are started as threads, they are threads of a channel initiator.)
   ```
   START CHANNEL(QM1.TO.QM2)
   ```
2. Use the control command runmqchl to start the channel as a process.
   ```
   runmqchl -c QM1.TO.QM2 -m QM1
   ```
3. Use the channel initiator to trigger the channel.

Stopping a channel:

Use the MQSC command STOP CHANNEL to request the channel to stop activity. The channel does not start a new batch of messages until the operator starts the channel again.

For information about restarting stopped channels, see “Restarting stopped channels” on page 797.

This command can be issued to a channel of any type except MQCHT_CLNTCONN.

You can select the type of stop you require:

Stop quiesce example

```
STOP CHANNEL(QM1.TO.QM2) MODE(QUIESCE)
```

This command requests the channel to close down in an orderly way. The current batch of messages is completed and the sync point procedure is carried out with the other end of the channel. If the channel is idle this command does not terminate a receiving channel.

Stop force example

```
STOP CHANNEL(QM1.TO.QM2) MODE(FORCE)
```

This option stops the channel immediately, but does not terminate the channel's thread or process. The channel does not complete processing the current batch of messages, and can, therefore, leave the channel in doubt. In general, consider using the quiesce stop option.

Stop terminate example

```
STOP CHANNEL(QM1.TO.QM2) MODE(TERMINATE)
```

This option stops the channel immediately, and terminates the channel's thread or process.

Stop (quiesce) stopped example

```
STOP CHANNEL(QM1.TO.QM2) STATUS(STOPPED)
```

This command does not specify a MODE, so defaults to MODE(QUIESCE). It requests that the channel is stopped so that it cannot be restarted automatically but must be started manually.

Stop (quiesce) inactive example

```
STOP CHANNEL(QM1.TO.QM2) STATUS(INACTIVE)
```

This command does not specify a MODE, so defaults to MODE(QUIESCE). It requests that the channel is made inactive so that it restarts automatically when required.

Renaming a channel:

Use MQSC to rename a message channel.

Use MQSC to carry out the following steps:

1. Use STOP CHANNEL to stop the channel.
2. Use DEFINE CHANNEL to create a duplicate channel definition with the new name.
3. Use DISPLAY CHANNEL to check that it has been created correctly.
4. Use DELETE CHANNEL to delete the original channel definition.

If you decide to rename a message channel, remember that a channel has ***two*** channel definitions, one at each end. Make sure that you rename the channel at both ends at the same time.

Resetting a channel:

Use the MQSC command RESET CHANNEL to change the message sequence number.

The RESET CHANNEL command is available for any message channel, but not for MQI channels (client-connection or server-connection). The first message starts the new sequence the next time the channel is started.

If the command is issued on a sender or server channel, it informs the other side of the change when the channel is restarted.

Related concepts:

"Getting started with objects" on page 810
Channels must be defined, and their associated objects must exist and be available for use, before a channel can be started. This section shows you how.

"Channel control function" on page 787
The channel control function provides facilities for you to define, monitor, and control channels.

"Connecting applications using distributed queuing" on page 759
This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

Related information:

RESET CHANNEL

Resolving in-doubt messages on a channel:

Use the MQSC command RESOLVE CHANNEL when messages are held in-doubt by a sender or server. For example because one end of the link has terminated, and there is no prospect of it recovering.

The RESOLVE CHANNEL command accepts one of two parameters: BACKOUT or COMMIT. Backout restores messages to the transmission queue, while Commit discards them.

The channel program does not try to establish a session with a partner. Instead, it determines the logical unit of work identifier (LUWID) which represents the in-doubt messages. It then issues, as requested, either:

- BACKOUT to restore the messages to the transmission queue; or
- COMMIT to delete the messages from the transmission queue.

For the resolution to succeed:

- The channel must be inactive
- The channel must be in doubt
- The channel type must be sender or server
- A local channel definition must exist
- The local queue manager must be running

Related concepts:

"Getting started with objects" on page 810
Channels must be defined, and their associated objects must exist and be available for use, before a channel can be started. This section shows you how.

"Channel control function" on page 787
The channel control function provides facilities for you to define, monitor, and control channels.

"Connecting applications using distributed queuing" on page 759
This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

Related information:

RESOLVE CHANNEL

Setting up communication for Windows

When a distributed-queuing management channel is started, it tries to use the connection specified in the channel definition. For this to succeed, it is necessary for the connection to be defined and available. This section explains how to do this using one of the four forms of communication for WebSphere MQ for Windows systems.

You might find it helpful to refer to Example configuration - IBM WebSphere MQ for Windows.

For UNIX and Linux systems see "Setting up communication on UNIX and Linux systems" on page 826.

Deciding on a connection

Choose from the following four forms of communication for WebSphere MQ for Windows systems:

- "Defining a TCP connection on Windows"
- "Defining an LU 6.2 connection on Windows" on page 819
- "Defining a NetBIOS connection on Windows" on page 821
- "Defining an SPX connection on Windows" on page 824 (Windows XP and Windows 2003 Server only)

Each channel definition must specify only one protocol as the Transmission protocol (Transport Type) attribute. One or more protocols can be used by a queue manager.

For WebSphere MQ clients, it might be useful to have alternative channels using different transmission protocols. For more information about WebSphere MQ clients, see Overview of clients.

Related concepts:

"Connecting applications using distributed queuing" on page 759
This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

"Monitoring and controlling channels on Windows, UNIX and Linux platforms" on page 807
For DQM you need to create, monitor, and control the channels to remote queue managers. You can control channels using commands, programs, WebSphere MQ Explorer, files for the channel definitions, and a storage area for synchronization information.

"Configuring connections between the server and client" on page 832
To configure the communication links between WebSphere MQ MQI clients and servers, decide on your communication protocol, define the connections at both ends of the link, start a listener, and define channels.

Defining a TCP connection on Windows:

Define a TCP connection by configuring a channel at the sending end to specify the address of the target, and by running a listener program at the receiving end.

Sending end

Specify the host name, or the TCP address of the target machine, in the Connection name field of the channel definition.

The port to connect to defaults to 1414. Port number 1414 is assigned by the Internet Assigned Numbers Authority to IBM WebSphere MQ.

To use a port number other than the default, specify it in the connection name field of the channel object definition thus:

```
DEFINE CHANNEL('channel name') CHLTYPE(SDR) +
         TRPTYPE(TCP) +
         CONNAME('OS2ROG3(1822)') +
         XMITQ('XMitQ name') +
         REPLACE
```

where `OS2ROG3` is the DNS name of the remote queue manager and `1822` is the port required. (This must be the port that the listener at the receiving end is listening on.)

A running channel must be stopped and restarted to pick up any change to the channel object definition.

You can change the default port number by specifying it in the `.ini` file for IBM WebSphere MQ for Windows:

```
TCP:
Port=1822
```

Note: To select which TCP/IP port number to use, IBM WebSphere MQ uses the first port number it finds in the following sequence:
1. The port number explicitly specified in the channel definition or command line. This number allows the default port number to be overridden for a channel.
2. The port attribute specified in the TCP stanza of the `.ini` file. This number allows the default port number to be overridden for a queue manager.
3. The default value of 1414. This is the number assigned to IBM WebSphere MQ by the Internet Assigned Numbers Authority for both inbound and outbound connections.

For more information about the values you set using qm.ini, see Configuration file stanzas for distributed queuing.

Receiving on TCP

To start a receiving channel program, a listener program must be started to detect incoming network requests and start the associated channel. You can use the IBM WebSphere MQ listener.

Receiving channel programs are started in response to a startup request from the sending channel.

To start a receiving channel program, a listener program must be started to detect incoming network requests and start the associated channel. You can use the IBM WebSphere MQ listener.

To run the Listener supplied with IBM WebSphere MQ, that starts new channels as threads, use the runmqlsr command.

A basic example of using the **`runmqlsr`** command:

```
runmqlsr -t tcp [-m QMNAME] [-p 1822]
```

The square brackets indicate optional parameters; QMNAME is not required for the default queue manager, and the port number is not required if you are using the default (1414). The port number must not exceed 65535.

Note: To select which TCP/IP port number to use, IBM WebSphere MQ uses the first port number it finds in the following sequence:

1. The port number explicitly specified in the channel definition or command line. This number allows the default port number to be overridden for a channel.
2. The port attribute specified in the TCP stanza of the .ini file. This number allows the default port number to be overridden for a queue manager.
3. The default value of 1414. This is the number assigned to IBM WebSphere MQ by the Internet Assigned Numbers Authority for both inbound and outbound connections.

For the best performance, run the IBM WebSphere MQ listener as a trusted application as described in "Running channels and listeners as trusted applications" on page 806. See Restrictions for trusted applications for information about trusted applications

Using the TCP/IP SO_KEEPALIVE option

If you want to use the Windows SO_KEEPALIVE option you must add the following entry to your registry:

```
TCP:
KeepAlive=yes
```

For more information about the SO_KEEPALIVE option, see "Checking that the other end of the channel is still available" on page 794.

On Windows, the HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters registry value for the Windows KeepAliveTime option controls the interval that elapses before the connection is checked. The default is two hours.

Defining an LU 6.2 connection on Windows:

SNA must be configured so that an LU 6.2 conversation can be established between the two machines.

Once the SNA is configured, proceed as follows.

See the following table for information.

Table 111. Settings on the local Windows system for a remote queue manager platform

Remote platform	TPNAME	TPPATH
z/OS or MVS/ESA without CICS	The same as in the corresponding side information about the remote queue manager.	-
z/OS or MVS/ESA using CICS	CKRC (sender) CKSV (requester) CKRC (server)	-
IBM i	The same as the compare value in the routing entry on the IBM i system.	-
UNIX and Linux systems	The same as in the corresponding side information about the remote queue manager.	*MQ_INSTALLATION_PATH*/bin/amqcrs6a

Table 111. Settings on the local Windows system for a remote queue manager platform (continued)

Remote platform	TPNAME	TPPATH
Windows	As specified in the Windows Run Listener command, or the invokable Transaction Program that was defined using TpSetup on Windows.	`MQ_INSTALLATION_PATH`\bin\amqcrs6a

`MQ_INSTALLATION_PATH` represents the high-level directory in which WebSphere MQ is installed.

If you have more than one queue manager on the same machine, ensure that the TPnames in the channel definitions are unique.

For the latest information about configuring AnyNet® SNA over TCP/IP, see the following online IBM documentation: AnyNet SNA over TCP/IP and SNA Node Operations.

Related concepts:

"Sending end on LU 6.2"
Create a CPI-C side object (symbolic destination) from the administration application of the LU 6.2 product you are using. Enter this name in the Connection name field in the channel definition. Also create an LU 6.2 link to the partner.

"Receiving on LU 6.2"
Receiving channel programs are started in response to a startup request from the sending channel.

Sending end on LU 6.2:

Create a CPI-C side object (symbolic destination) from the administration application of the LU 6.2 product you are using. Enter this name in the Connection name field in the channel definition. Also create an LU 6.2 link to the partner.

In the CPI-C side object enter the partner LU Name at the receiving machine, the TP Name and the Mode Name. For example:

```
Partner LU Name             OS2ROG2
Partner TP Name             recv
Mode Name                   #INTER
```

Receiving on LU 6.2:

Receiving channel programs are started in response to a startup request from the sending channel.

To start a receiving channel program, a listener program has to be started to detect incoming network requests and start the associated channel. You start this listener program with the RUNMQLSR command, giving the TpName to listen on. Alternatively, you can use TpStart under SNA Server for Windows.

Using the RUNMQLSR command

Example of the command to start the listener:

```
RUNMQLSR -t LU62 -n RECV [-m QMNAME]
```

where `RECV` is the TpName that is specified at the other (sending) end as the "TpName to start on the remote side". The last part in square brackets is optional and is not required for the default queue manager.

It is possible to have more than one queue manager running on one machine. You must assign a different TpName to each queue manager, and then start a listener program for each one. For example:

```
RUNMQLSR -t LU62 -m QM1 -n TpName1
RUNMQLSR -t LU62 -m QM2 -n TpName2
```

For the best performance, run the WebSphere MQ listener as a trusted application as described in Running channels and listeners as trusted applications. See Restrictions for trusted applications for information about trusted applications.

You can stop all WebSphere MQ listeners running on a queue manager that is inactive, using the command:

```
ENDMQLSR [-m QMNAME]
```

Using Microsoft SNA Server on Windows

You can use TpSetup (from the SNA Server SDK) to define an invokable TP that then drives amqcrs6a.exe, or you can set various registry values manually. The parameters that should be passed to amqcrs6a.exe are:

```
  -m QM -n TpName
```

where `QM` is the Queue Manager name and `TpName` is the TP Name. See the *Microsoft SNA Server APPC Programmers Guide* or the *Microsoft SNA Server CPI-C Programmers Guide* for more information.

If you do not specify a queue manager name, the default queue manager is assumed.

Defining a NetBIOS connection on Windows:

WebSphere MQ uses three types of NetBIOS resource when establishing a NetBIOS connection to another WebSphere MQ product: sessions, commands, and names. Each of these resources has a limit, which is established either by default or by choice during the installation of NetBIOS.

Each running channel, regardless of type, uses one NetBIOS session and one NetBIOS command. The IBM NetBIOS implementation allows multiple processes to use the same local NetBIOS name. Therefore, only one NetBIOS name needs to be available for use by WebSphere MQ. Other vendors' implementations, for example Novell's NetBIOS emulation, require a different local name per process. Verify your requirements from the documentation for the NetBIOS product you are using.

In all cases, ensure that sufficient resources of each type are already available, or increase the maximums specified in the configuration. Any changes to the values require a system restart.

During system startup, the NetBIOS device driver displays the number of sessions, commands, and names available for use by applications. These resources are available to any NetBIOS-based application that is running on the same system. Therefore, it is possible for other applications to consume these resources before WebSphere MQ needs to acquire them. Your LAN network administrator should be able to clarify this for you.

Related concepts:

"Defining the IBM WebSphere MQ local NetBIOS name" on page 822
The local NetBIOS name used by IBM WebSphere MQ channel processes can be specified in three ways.

"Establishing the queue manager NetBIOS session, command, and name limits" on page 822
The queue manager limits for NetBIOS sessions, commands, and names can be specified in two ways.

"Establishing the LAN adapter number" on page 823
For channels to work successfully across NetBIOS, the adapter support at each end must be compatible. IBM WebSphere MQ allows you to control the choice of LAN adapter (LANA) number by using the AdapterNum value in the NETBIOS stanza of your qm.ini file and by specifying the `-a` parameter on the runmqlsr command.

"Initiating the NetBIOS connection" on page 823
Defining the steps needed to initiate a connection.

"Target listener for the NetBIOS connection" on page 823
Defining the steps to be undertaken at the receiving end of the NetBIOS connection.

Defining the IBM WebSphere MQ local NetBIOS name:

The local NetBIOS name used by IBM WebSphere MQ channel processes can be specified in three ways.

In order of precedence the three ways are:

1. The value specified in the `-l` parameter of the RUNMQLSR command, for example:
   ```
   RUNMQLSR -t NETBIOS -l my_station
   ```
2. The MQNAME environment variable with a value that is established by the command:
   ```
   SET MQNAME=my_station
   ```
   You can set the MQNAME value for each process. Alternatively, you can set it at a system level in the Windows registry.

   If you are using a NetBIOS implementation that requires unique names, you must issue a SET MQNAME command in each window in which a IBM WebSphere MQ process is started. The MQNAME value is arbitrary but it must be unique for each process.
3. The NETBIOS stanza in the queue manager configuration file qm.ini. For example:
   ```
   NETBIOS:

     LocalName=my_station
   ```

Note:

1. Due to the variations in implementation of the NetBIOS products supported, you are advised to make each NetBIOS name unique in the network. If you do not, unpredictable results might occur. If you have problems establishing a NetBIOS channel and there are error messages in the queue-manager error log showing a NetBIOS return code of X'15', review your use of NetBIOS names.
2. On Windows, you cannot use your machine name as the NetBIOS name because Windows already uses it.
3. Sender channel initiation requires that a NetBIOS name be specified either by using the MQNAME environment variable or the LocalName in the qm.ini file.

Establishing the queue manager NetBIOS session, command, and name limits:

The queue manager limits for NetBIOS sessions, commands, and names can be specified in two ways.

In order of precedence these ways are:

1. The values specified in the RUNMQLSR command:
   ```
   -s  Sessions
   -e  Names
   -o  Commands
   ```
   If the -m operand is not specified in the command, the values apply only to the default queue manager.
2. The NETBIOS stanza in the queue manager configuration file qm.ini. For example:
   ```
   NETBIOS:

     NumSess=Qmgr_max_sess
     NumCmds=Qmgr_max_cmds
     NumNames=Qmgr_max_names
   ```

Establishing the LAN adapter number:

For channels to work successfully across NetBIOS, the adapter support at each end must be compatible. IBM WebSphere MQ allows you to control the choice of LAN adapter (LANA) number by using the AdapterNum value in the NETBIOS stanza of your qm.ini file and by specifying the `-a` parameter on the runmqlsr command.

The default LAN adapter number used by IBM WebSphere MQ for NetBIOS connections is 0. Verify the number being used on your system as follows:

On Windows, it is not possible to query the LAN adapter number directly through the operating system. Instead, you use the LANACFG.EXE command-line utility, available from Microsoft. The output of the tool shows the virtual LAN adapter numbers and their effective bindings. For further information about LAN adapter numbers, see the Microsoft Knowledge Base article 138037 *HOWTO: Use LANA Numbers in a 32-bit Environment*.

Specify the correct value in the NETBIOS stanza of the queue manager configuration file, qm.ini:

```
NETBIOS:
  AdapterNum=n
```

where n is the correct LAN adapter number for this system.

Initiating the NetBIOS connection:

Defining the steps needed to initiate a connection.

To initiate the connection, follow these steps at the sending end:

1. Define the NetBIOS station name using the MQNAME or LocalName value.
2. Verify the LAN adapter number being used on your system and specify the correct file using the AdapterNum.
3. In the ConnectionName field of the channel definition, specify the NetBIOS name being used by the target listener program. On Windows, NetBIOS channels ***must*** be run as threads. Do this by specifying MCATYPE(THREAD) in the channel definition.

```
DEFINE CHANNEL (chname) CHLTYPE(SDR) +
       TRPTYPE(NETBIOS) +
       CONNAME(your_station) +
       XMITQ(xmitq) +
       MCATYPE(THREAD) +
       REPLACE
```

Target listener for the NetBIOS connection:

Defining the steps to be undertaken at the receiving end of the NetBIOS connection.

At the receiving end, follow these steps:

1. Define the NetBIOS station name using the MQNAME or LocalName value.
2. Verify the LAN adapter number being used on your system and specify the correct file using the AdapterNum.
3. Define the receiver channel:

```
DEFINE CHANNEL (chname) CHLTYPE(RCVR) +
       TRPTYPE(NETBIOS) +
       REPLACE
```

4. Start the WebSphere MQ listener program to establish the station and make it possible to contact it. For example:

```
RUNMQLSR -t NETBIOS -l your_station [-m qmgr]
```

This command establishes `your_station` as a NetBIOS station waiting to be contacted. The NetBIOS station name must be unique throughout your NetBIOS network.

For the best performance, run the WebSphere MQ listener as a trusted application as described in "Running channels and listeners as trusted applications" on page 806. See Restrictions for trusted applications for information about trusted applications.

You can stop all WebSphere MQ listeners running on a queue manager that is inactive, using the command:

```
ENDMQLSR [-m QMNAME]
```

If you do not specify a queue manager name, the default queue manager is assumed.

Defining an SPX connection on Windows:

An SPX connection applies only to a client and server running Windows XP and Windows 2003 Server.

The channel definition at the sending end specifies the address of the target. A listener program must be run at the receiving end.

Related concepts:

"Sending end on SPX"
If the target machine is remote, specify the SPX address of the target machine in the Connection name field of the channel definition.

"Receiving on SPX" on page 825
Receiving channel programs are started in response to a startup request from the sending channel.

"IPX/SPX parameters" on page 826
In most cases, the default settings for the IPX/SPX parameters will suit your needs. However, you might need to modify some of them in your environment to tune its use for WebSphere MQ.

Sending end on SPX:

If the target machine is remote, specify the SPX address of the target machine in the Connection name field of the channel definition.

The SPX address is specified in the following form:

```
network.node(socket)
```

where:

network
Is the 4-byte network address of the network on which the remote machine resides,

node Is the 6-byte node address, which is the LAN address of the LAN adapter in the remote machine

socket Is the 2-byte socket number on which the remote machine listens.

If the local and remote machines are on the same network then the network address need not be specified. If the remote end is listening on the default socket (5E86) then the socket need not be specified.

An example of a fully specified SPX address specified in the CONNAME parameter of an MQSC command is:

```
CONNAME('00000001.08005A7161E5(5E87)')
```

In the default case, where the machines are both on the same network, this becomes:

```
CONNAME(08005A7161E5)
```

The default socket number can be changed by specifying it in the queue manager configuration file (qm.ini):

```
SPX:
  Socket=5E87
```

For more information about the values you set using qm.ini, see Configuration file stanzas for distributed queuing.

Receiving on SPX:

Receiving channel programs are started in response to a startup request from the sending channel.

To start a receiving channel program, a listener program must be started to detect incoming network requests and start the associated channel.

Use the WebSphere MQ listener.

Using the SPX listener backlog option

When receiving on SPX, a maximum number of outstanding connection requests is set. This can be considered a *backlog* of requests waiting on the SPX port for the listener to accept the request. The default listener backlog values are shown in Table 112.

Table 112. Default outstanding connection requests on Windows

Platform	Default listener backlog value
Windows Server	5
Windows Workstation	5

If the backlog reaches the values in Table 112, the reason code, MQRC_Q_MGR_NOT_AVAILABLE is received when trying to connect to the queue manager using MQCONN or MQCONNX. If this happens, it is possible to try to connect again.

However, to avoid this error, you can add an entry in the qm.ini file or in the registry for Windows:

```
SPX:
ListenerBacklog = n
```

This overrides the default maximum number of outstanding requests (see Table 112) for the SPX listener.

Note: Some operating systems support a larger value than the default. If necessary, this can be used to avoid reaching the connection limit.

To run the listener with the `backlog` option switched on either:

- Use the `RUNMQLSR -b` command, or
- Use the MQSC command **`DEFINE LISTENER`** with the BACKLOG attribute set to the required value.

For information about the **`RUNMQLSR`** command, see runmqlsr. For information about the DEFINE LISTENER command, see DEFINE LISTENER.

Using the WebSphere MQ listener

To run the Listener supplied with WebSphere MQ, that starts new channels as threads, use the RUNMQLSR command. For example:

```
RUNMQLSR -t spx [-m QMNAME] [-x 5E87]
```

The square brackets indicate optional parameters; QMNAME is not required for the default queue manager, and the socket number is not required if you are using the default (5E86).

For the best performance, run the WebSphere MQ listener as a trusted application as described in "Running channels and listeners as trusted applications" on page 806. See Restrictions for trusted applications for more information about trusted applications.

You can stop all WebSphere MQ listeners running on a queue manager that is inactive, using the command:

```
ENDMQLSR [-m QMNAME]
```

If you do not specify a queue manager name, the default queue manager is assumed.

IPX/SPX parameters:

In most cases, the default settings for the IPX/SPX parameters will suit your needs. However, you might need to modify some of them in your environment to tune its use for WebSphere MQ.

The actual parameters and the method of changing them varies according to the platform and provider of SPX communications support. The example section describes some of these parameters, particularly those that might influence the operation of WebSphere MQ channels and client connections.

Windows systems

Refer to the Microsoft documentation for full details of the use and setting of the NWLink IPX and SPX parameters. The IPX/SPX parameters are in the following paths in the registry:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Service\NWLinkSPX\Parameters
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Service\NWLinkIPX\Parameters
```

Setting up communication on UNIX and Linux systems

DQM is a remote queuing facility for WebSphere MQ. It provides channel control programs for the queue manager which form the interface to communication links, controllable by the system operator. The channel definitions held by distributed-queuing management use these connections.

When a distributed-queuing management channel is started, it tries to use the connection specified in the channel definition. To succeed, it is necessary for the connection to be defined and available. This section explains how to do this. You might also find it helpful to refer to the following sections:

- Example configuration - IBM WebSphere MQ for AIX
- Example configuration - IBM WebSphere MQ for HP-UX
- Example configuration - IBM WebSphere MQ for Solaris
- Example configuration - IBM WebSphere MQ for Linux

For Windows, see "Setting up communication for Windows" on page 817.

You can choose between two forms of communication for WebSphere MQ on UNIX and Linux systems:

- "Defining a TCP connection on UNIX and Linux" on page 827
- "Defining an LU 6.2 connection on UNIX and Linux" on page 830

Each channel definition must specify one only as the transmission protocol (Transport Type) attribute. One or more protocols can be used by a queue manager.

For WebSphere MQ MQI clients, it might be useful to have alternative channels using different transmission protocols. For more information about WebSphere MQ MQI clients, see Overview of IBM WebSphere MQ MQI clients.

Related concepts:

"Connecting applications using distributed queuing" on page 759
This section provides more detailed information about intercommunication between WebSphere MQ installations, including queue definition, channel definition, triggering, and sync point procedures

"Monitoring and controlling channels on Windows, UNIX and Linux platforms" on page 807
For DQM you need to create, monitor, and control the channels to remote queue managers. You can control channels using commands, programs, WebSphere MQ Explorer, files for the channel definitions, and a storage area for synchronization information.

"Configuring connections between the server and client" on page 832
To configure the communication links between WebSphere MQ MQI clients and servers, decide on your communication protocol, define the connections at both ends of the link, start a listener, and define channels.

Defining a TCP connection on UNIX and Linux:

The channel definition at the sending end specifies the address of the target. The listener or inet daemon is configured for the connection at the receiving end.

Sending end

Specify the host name, or the TCP address of the target machine, in the Connection Name field of the channel definition. The port to connect to defaults to 1414. Port number 1414 is assigned by the Internet Assigned Numbers Authority to WebSphere MQ.

To use a port number other than the default, change the connection name field thus:

```
Connection Name REMHOST(1822)
```

where REMHOST is the host name of the remote machine and 1822 is the port number required. (This must be the port that the listener at the receiving end is listening on.)

Alternatively you can change the port number by specifying it in the queue manager configuration file (qm.ini):

```
TCP:
  Port=1822
```

For more information about the values you set using qm.ini, see Configuration file stanzas for distributed queuing.

Receiving on TCP

You can use either the TCP/IP listener, which is the inet daemon (inetd), or the WebSphere MQ listener.

Some Linux distributions now use the extended inet daemon (xinetd) instead of the inet daemon. For information about how to use the extended inet daemon on a Linux system, see Establishing a TCP connection on Linux.

Related concepts:

"Using the TCP/IP listener"
To start channels on UNIX and Linux, the /etc/services file and the inetd.conf file must be edited

"Using the TCP listener backlog option" on page 829
In TCP, connections are treated incomplete unless three-way handshake takes place between the server and the client. These connections are called outstanding connection requests. A maximum value is set for these outstanding connection requests and can be considered a backlog of requests waiting on the TCP port for the listener to accept the request.

"Using the WebSphere MQ listener" on page 830
To run the listener supplied with WebSphere MQ, which starts new channels as threads, use the `runmqlsr` command.

"Using the TCP/IP SO_KEEPALIVE option" on page 830
On some UNIX and Linux systems, you can define how long TCP waits before checking that the connection is still available, and how frequently it tries the connection again if the first check fails. This is either a kernel tunable parameter, or can be entered at the command line.

Using the TCP/IP listener:

To start channels on UNIX and Linux, the /etc/services file and the inetd.conf file must be edited

Follow these instructions:

1. Edit the /etc/services file:

   Note: To edit the /etc/services file, you must be logged in as a superuser or root. You can change this, but it must match the port number specified at the sending end.
   Add the following line to the file:

   ```
   MQSeries 1414/tcp
   ```

   where 1414 is the port number required by WebSphere MQ. The port number must not exceed 65535.
2. Add a line in the inetd.conf file to call the program amqcrsta, where *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed:

   ```
   MQSeries stream tcp nowait mqm MQ_INSTALLATION_PATH/bin/amqcrsta amqcrsta
   [-m Queue_Man_Name]
   ```

The updates are active after inetd has reread the configuration files. To do this, issue the following commands from the root user ID:

- On AIX:
  ```
  refresh -s inetd
  ```
- On HP-UX, from the mqm user ID:
  ```
  inetd -c
  ```
- On Solaris 10 or later:
  ```
  inetconv
  ```
- On other UNIX and Linux systems (including Solaris 9):
  ```
  kill -1 <process number>
  ```

When the listener program started by inetd inherits the locale from inetd, it is possible that the MQMDE is not honored (merged) and is placed on the queue as message data. To ensure that the MQMDE is honored, you must set the locale correctly. The locale set by inetd might not match that chosen for other locales used by WebSphere MQ processes. To set the locale:

1. Create a shell script which sets the locale environment variables LANG, LC_COLLATE, LC_CTYPE, LC_MONETARY, LC_NUMERIC, LC_TIME, and LC_MESSAGES to the locale used for other WebSphere MQ process.
2. In the same shell script, call the listener program.

3. Modify the inetd.conf file to call your shell script in place of the listener program.

It is possible to have more than one queue manager on the server. You must add a line to each of the two files, for each of the queue managers. For example:

```
MQSeries1     1414/tcp
MQSeries2     1822/tcp
```

```
MQSeries2 stream tcp nowait mqm MQ_INSTALLATION_PATH/bin/amqcrsta amqcrsta -m QM2
```

Where *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed.

This avoids error messages being generated if there is a limitation on the number of outstanding connection requests queued at a single TCP port. For information about the number of outstanding connection requests, see "Using the TCP listener backlog option."

Using the TCP listener backlog option:

In TCP, connections are treated incomplete unless three-way handshake takes place between the server and the client. These connections are called outstanding connection requests. A maximum value is set for these outstanding connection requests and can be considered a backlog of requests waiting on the TCP port for the listener to accept the request.

The default listener backlog values are shown in Table 113.

Table 113. Maximum outstanding connection requests queued at a TCP/IP port

Server platform	Maximum connection requests
AIX	100
HP-UX	20
Linux	100
IBM i	255
Solaris	100
Windows Server	100
Windows Workstation	100
z/OS	255

If the backlog reaches the values shown in Table 113, the TCP/IP connection is rejected and the channel is not able to start.

For MCA channels, this results in the channel going into a RETRY state and trying the connection again at a later time.

However, to avoid this error, you can add an entry in the qm.ini file:

```
TCP:
ListenerBacklog = n
```

This overrides the default maximum number of outstanding requests (see Table 113) for the TCP/IP listener.

Note: Some operating systems support a larger value than the default. If necessary, this value can be used to avoid reaching the connection limit.

To run the listener with the `backlog` option switched on either:

- Use the `runmqlsr -b` command, or

- Use the MQSC command **DEFINE LISTENER** with the BACKLOG attribute set to the required value.

For information about the **runmqlsr** command, see runmqlsr. For information about the DEFINE LISTENER command, see the DEFINE LISTENER.

Using the WebSphere MQ listener:

To run the listener supplied with WebSphere MQ, which starts new channels as threads, use the `runmqlsr` command.

For example:

```
runmqlsr -t tcp [-m QMNAME] [-p 1822]
```

The square brackets indicate optional parameters; `QMNAME` is not required for the default queue manager, and the port number is not required if you are using the default (1414). The port number must not exceed 65535.

For the best performance, run the WebSphere MQ listener as a trusted application as described in “Running channels and listeners as trusted applications” on page 806. See Restrictions for trusted applications for information about trusted applications.

You can stop all WebSphere MQ listeners running on a queue manager that is inactive, using the command:

```
endmqlsr [-m QMNAME]
```

If you do not specify a queue manager name, the default queue manager is assumed.

Using the TCP/IP SO_KEEPALIVE option:

On some UNIX and Linux systems, you can define how long TCP waits before checking that the connection is still available, and how frequently it tries the connection again if the first check fails. This is either a kernel tunable parameter, or can be entered at the command line.

If you want to use the SO_KEEPALIVE option (for more information, see “Checking that the other end of the channel is still available” on page 794) you must add the following entry to your queue manager configuration file (qm.ini):

```
TCP:
   KeepAlive=yes
```

See the documentation for your UNIX and Linux system for more information.

Defining an LU 6.2 connection on UNIX and Linux:

SNA must be configured so that an LU 6.2 conversation can be established between the two machines.

For the latest information about configuring SNA over TCP/IP, see the following online IBM documentation: Communications Server.

SNA must be configured so that an LU 6.2 conversation can be established between the two systems.

See the *Multiplatform APPC Configuration Guide* and the following table for information.

Table 114. Settings on the local UNIX and Linux system for a remote queue manager platform

Remote platform	TPNAME	TPPATH
z/OS without CICS	The same as the corresponding TPName in the side information about the remote queue manager.	-
z/OS using CICS	CKRC (sender) CKSV (requester) CKRC (server)	-
IBM i	The same as the compare value in the routing entry on the IBM i system.	-
UNIX and Linux systems	The same as the corresponding TPName in the side information about the remote queue manager.	*MQ_INSTALLATION_PATH*/bin/amqcrs6a
Windows	As specified in the Windows Run Listener command, or the invokable Transaction Program that was defined using TpSetup on Windows.	*MQ_INSTALLATION_PATH*\bin\amqcrs6a

MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.

If you have more than one queue manager on the same machine, ensure that the TPnames in the channel definitions are unique.

Related concepts:

"Sending end"
On UNIX and Linux systems, create a CPI-C side object (symbolic destination) and enter this name in the Connection name field in the channel definition. Also create an LU 6.2 link to the partner.

"Receiving on LU 6.2" on page 832
On UNIX and Linux systems, create a listening attachment at the receiving end, an LU 6.2 logical connection profile, and a TPN profile.

Sending end:

On UNIX and Linux systems, create a CPI-C side object (symbolic destination) and enter this name in the Connection name field in the channel definition. Also create an LU 6.2 link to the partner.

In the CPI-C side object enter the partner LU name at the receiving machine, the transaction program name and the mode name. For example:

```
Partner LU Name             REMHOST
Remote TP Name              recv
Service Transaction Program no
Mode Name                   #INTER
```

On HP-UX, use the APPCLLU environment variable to name the local LU that the sender should use. On Solaris, set the APPC_LOCAL_LU environment variable to be the local LU name.

SECURITY PROGRAM is used, where supported by CPI-C, when WebSphere MQ attempts to establish an SNA session.

Receiving on LU 6.2:

On UNIX and Linux systems, create a listening attachment at the receiving end, an LU 6.2 logical connection profile, and a TPN profile.

In the TPN profile, enter the full path to the executable file and the Transaction Program name:

```
Full path to TPN executable   MQ_INSTALLATION_PATH/bin/amqcrs6a
Transaction Program name      recv
User ID                       0
```

MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.

On systems where you can set the user ID, specify a user who is a member of the mqm group. On AIX, Solaris, and HP-UX, set the APPCTPN (transaction name) and APPCLLU (local LU name) environment variables (you can use the configuration panels for the invoked transaction program).

You might need to use a queue manager other than the default queue manager. If so, define a command file that calls:

```
amqcrs6a -m Queue_Man_Name
```

then call the command file.

Configuring connections between the server and client

To configure the communication links between WebSphere MQ MQI clients and servers, decide on your communication protocol, define the connections at both ends of the link, start a listener, and define channels.

In WebSphere MQ, the logical communication links between objects are called *channels*. The channels used to connect WebSphere MQ MQI clients to servers are called MQI channels. You set up channel definitions at each end of your link so that your WebSphere MQ application on the WebSphere MQ MQI client can communicate with the queue manager on the server. For a detailed description of how to do this, see User defined channels.

Before you define your MQI channels, you must:

1. Decide on the form of communication you are going to use. See "Which communication type to use."
2. Define the connection at each end of the channel:

   To define the connection, you must:

   - Configure the connection.
   - Record the values of the parameters that you need for the channel definitions.
   - Enable the server to detect incoming network requests from your WebSphere MQ MQI client, by starting a *listener*.

Which communication type to use

Different platforms support different transmission protocols. Your choice of transmission protocol depends on your combination of WebSphere MQ MQI client and server platforms.

There are up to four types of transmission protocol for MQI channels depending on your client and server platforms:

- LU 6.2
- NetBIOS
- SPX

- TCP/IP

When you define your MQI channels, each channel definition must specify a transmission protocol (transport type) attribute. A server is not restricted to one protocol, so different channel definitions can specify different protocols. For WebSphere MQ MQI clients, it might be useful to have alternative MQI channels using different transmission protocols.

Your choice of transmission protocol might be restricted by your particular combination of WebSphere MQ MQI client and server platforms. The possible combinations are shown in the following table.

Table 115. Transmission protocols - combination of WebSphere MQ MQI client and server platforms

Transmission protocol	**WebSphere MQ MQI client**	**WebSphere MQ server**
TCP/IP	UNIX systems Windows	UNIX systems Windows z/OS
LU 6.2	UNIX systems[1] Windows	UNIX systems[1] Windows
NetBIOS	Windows	Windows
SPX	Windows	Windows
Note: 1. Except Linux for Power Systems™		

For more about setting up different types of connections, see the following links:
- “Defining a TCP connection on Windows” on page 817
- “Defining a TCP connection on UNIX and Linux” on page 827
- “TCP/IP connection limits” on page 835
- “Defining an LU 6.2 connection on Windows” on page 819
- “Defining an LU 6.2 connection on UNIX and Linux” on page 830
- “Defining a NetBIOS connection on Windows” on page 821
- “Defining an SPX connection on Windows” on page 824

Related concepts:

“Configuring an extended transactional client” on page 836
This collection of topics describes how to configure the extended transactional function for each category of transaction manager.

“Defining MQI channels” on page 846
To create a new channel, you have to create **two** channel definitions, one for each end of the connection, using the same channel name and compatible channel types. In this case, the channel types are *server-connection* and *client-connection*.

“Creating server-connection and client-connection definitions on different platforms” on page 847
You can create each channel definition on the computer to which it applies. There are restrictions on how you can create channel definitions on a client computer.

“Creating server-connection and client-connection definitions on the server” on page 850
You can create both definitions on the server, then make the client-connection definition available to the client.

"Channel-exit programs for MQI channels" on page 855
Three types of channel exit are available to the WebSphere MQ MQI client environment on UNIX, Linux and Windows systems.

"Connecting a client to a queue-sharing group" on page 859
You can connect a client to a queue-sharing group by creating an MQI channel between a client and a queue manager on a server that is a member of a queue-sharing group.

"Configuring a client using a configuration file" on page 860
Configure your clients using attributes in a text file. These attributes can be overridden by environment variables or in other platform-specific ways.

Related information:

Connecting IBM MQ MQI client applications to queue managers

DISPLAY CHLAUTH

SET CHLAUTH

Which communication type to use

Different platforms support different communication protocols. Your choice of transmission protocol depends on your combination of WebSphere MQ MQI client and server platforms.

There are four types of communication for MQI channels on different platforms:

- LU 6.2
- NetBIOS
- SPX
- TCP/IP

When you define your MQI channels, each channel definition must specify a transmission protocol (transport type) attribute. A server is not restricted to one protocol, so different channel definitions can specify different protocols. For WebSphere MQ MQI clients, it might be useful to have alternative MQI channels using different transmission protocols.

Your choice of transmission protocol also depends on your particular combination of WebSphere MQ client and server platforms. The possible combinations are shown in the following table.

Table 116. Transmission protocols - combination of WebSphere MQ client and server platforms

Transmission protocol	**WebSphere MQ MQI client**	**WebSphere MQ server**
TCP/IP	UNIX systems Windows	UNIX systems Windows
LU 6.2	UNIX systems[1] Windows	UNIX systems[1] Windows
NetBIOS	Windows	Windows
SPX	Windows	Windows
Note: 1. Except Linux (POWER® platform)		

Related concepts:

"Defining a TCP connection on Windows" on page 817
Define a TCP connection by configuring a channel at the sending end to specify the address of the target, and by running a listener program at the receiving end.

"Defining a TCP connection on UNIX and Linux" on page 827
The channel definition at the sending end specifies the address of the target. The listener or inet daemon is configured for the connection at the receiving end.

"Defining an LU 6.2 connection on Windows" on page 819
SNA must be configured so that an LU 6.2 conversation can be established between the two machines.

"Defining an LU 6.2 connection on UNIX and Linux" on page 830
SNA must be configured so that an LU 6.2 conversation can be established between the two machines.

"Defining a NetBIOS connection on Windows" on page 821
WebSphere MQ uses three types of NetBIOS resource when establishing a NetBIOS connection to another WebSphere MQ product: sessions, commands, and names. Each of these resources has a limit, which is established either by default or by choice during the installation of NetBIOS.

"Defining an SPX connection on Windows" on page 824
An SPX connection applies only to a client and server running Windows XP and Windows 2003 Server.

Related reference:

"TCP/IP connection limits"
The number of outstanding connection requests that can be queued at a single TCP/IP port depends on the platform. An error occurs if the limit is reached.

Defining a TCP/IP connection

Specifying a transport type of TCP on the channel definition at the client end. Start a listener program on the server.

Specify a TCP/IP connection at the client by specifying a transport type of TCP on the channel definition.

Receiving channel programs are started in response to a startup request from the sending channel. To do this, a listener program has to be started to detect incoming network requests and start the associated channel. The procedure for starting a listener program depends on the server platform.

See the related topics for your client and server platforms.

TCP/IP connection limits

The number of outstanding connection requests that can be queued at a single TCP/IP port depends on the platform. An error occurs if the limit is reached.

This connection limit is not the same as the maximum number of clients you can attach to a IBM WebSphere MQ server. You can connect more clients to a server, up to the level determined by the server system resources. The backlog values for connection requests are shown in the following table:

Table 117. Maximum outstanding connection requests queued at a TCP/IP port

Server platform	Maximum connection requests
AIX	100
HP-UX	20
Linux	100
IBM	255
Solaris	100
Windows Server	100
Windows Workstation	100

Table 117. Maximum outstanding connection requests queued at a TCP/IP port (continued)

Server platform	Maximum connection requests
z/OS	255

If the connection limit is reached, the client receives a return code of MQRC_HOST_NOT_AVAILABLE from the **MQCONN** call, and an AMQ9202 error in the client error log (`/var/mqm/errors/AMQERR0n.LOG` on UNIX and Linux systems or `amqerr0n.log` in the errors subdirectory of the IBM WebSphere MQ client installation on Windows). If the client retries the **MQCONN** request, it might be successful.

To increase the number of connection requests you can make, and avoid error messages being generated by this limitation, you can have multiple listeners each listening on a different port, or have more than one queue manager.

Defining a NetBIOS or SPX connection

NetBIOS and SPX connections apply only to Windows systems.

A NetBIOS connection applies only to a client and server running Windows. See Defining a NetBIOS connection.

An SPX connection applies only to a client and server running Windows XP or Windows 2003 Server. See Defining an SPX connection.

Configuring an extended transactional client

This collection of topics describes how to configure the extended transactional function for each category of transaction manager.

For each platform, the extended transactional client provides support for the following external transaction managers:

XA-compliant transaction managers
The extended transactional client provides the XA resource manager interface to support XA-compliant transaction managers such as CICS and Tuxedo.

Microsoft Transaction Server (Windows systems only)
On Windows systems only, the XA resource manager interface also supports Microsoft Transaction Server (MTS). The WebSphere MQ MTS support supplied with the extended transactional client provides the bridge between MTS and the XA resource manager interface.

WebSphere Application Server
Earlier versions of WebSphere MQ supported WebSphere Application Server Version 4 or Version 5, and required you to carry out certain configuration tasks to use the extended transactional client. WebSphere Application Server Version 6 and later includes a WebSphere MQ messaging provider, so you do not need to use the extended transactional client.

Related concepts:

"Configuring XA-compliant transaction managers"
First configure the WebSphere MQ base client, then configure the extended transactional function using the information in these topics.

"Microsoft Transaction Server" on page 845
No additional configuration is required before you can use MTS as a transaction manager. However, there are some points to note.

Configuring XA-compliant transaction managers

First configure the WebSphere MQ base client, then configure the extended transactional function using the information in these topics.

Note: This section assumes that you have a basic understanding of the XA interface as published by The Open Group in *Distributed Transaction Processing: The XA Specification*.

To configure an extended transactional client, you must first configure the WebSphere MQ base client as described in Installing a IBM WebSphere MQ client. Using the information in this section, you can then configure the extended transactional function for an XA-compliant transaction manager such as CICS and Tuxedo.

A transaction manager communicates with a queue manager as a resource manager using the same MQI channel as that used by the client application that is connected to the queue manager. When the transaction manager issues a resource manager (xa_) function call, the MQI channel is used to forward the call to the queue manager, and to receive the output back from the queue manager.

Either the transaction manager can start the MQI channel by issuing an xa_open call to open the queue manager as a resource manager, or the client application can start the MQI channel by issuing an MQCONN or MQCONNX call.

- If the transaction manager starts the MQI channel, and the client application later calls MQCONN or MQCONNX on the same thread, the MQCONN or MQCONNX call completes successfully and a connection handle is returned to the application. The application does not receive a MQCC_WARNING completion code with an MQRC_ALREADY_CONNECTED reason code.
- If the client application starts the MQI channel, and the transaction manager later calls xa_open on the same thread, the xa_open call is forwarded to the queue manager using the MQI channel.

In a recovery situation following a failure, when no client applications are running, the transaction manager can use a dedicated MQI channel to recover any incomplete units of work in which the queue manager was participating at the time of the failure.

Note the following conditions when using an extended transactional client with an XA-compliant transaction manager:

- Within a single thread, a client application can be connected to only one queue manager at a time. This restriction applies only when using an extended transactional client; a client application that is using a WebSphere MQ base client can be connected to more than one queue manager concurrently within a single thread.
- Each thread of a client application can connect to a different queue manager.
- A client application cannot use shared connection handles.

To configure the extended transactional function, you must provide the following information to the transaction manager for each queue manager that acts as a resource manager:

- An xa_open string
- A pointer to an XA switch structure

When the transaction manager calls xa_open to open the queue manager as a resource manager, it passes the xa_open string to the extended transactional client as the argument, *xa_info*, on the call. The extended transactional client uses the information in the xa_open string in the following ways:

- To start an MQI channel to the server queue manager, if the client application has not already started one
- To check that the queue manager that the transaction manager opens as a resource manager is the same as the queue manager to which the client application connects
- To locate the transaction manager's ax_reg and ax_unreg functions, if the queue manager uses dynamic registration

For the format of an xa_open string, and for more details about how the information in the xa_open string is used by an extended transactional client, see “The format of an xa_open string” on page 838.

An XA switch structure enables the transaction manager to locate the xa_ functions provided by the extended transactional client, and specifies whether the queue manager uses dynamic registration. For information about the XA switch structures supplied with an extended transactional client, see "The XA switch structures" on page 842.

For information about how to configure the extended transactional function for a particular transaction manager, and for any other information about using the transaction manager with an extended transactional client, see the following sections:

- "Configuring an extended transactional client for CICS" on page 843
- "Configuring an extended transactional client for Tuxedo" on page 845

Related concepts:

"The CHANNEL, TRPTYPE, CONNAME, and QMNAME parameters of the xa_open string" on page 840
Use this information to understand how the extended transactional client uses these parameters to determine the queue manager to connect to.

"Additional error processing for xa_open" on page 841
The xa_open call fails in certain circumstances.

Related tasks:

"Using the extended transactional client with SSL channels" on page 843
You cannot set up an SSL channel using the xa_open string. Follow these instructions to use the client channel definition table (ccdt).

Related reference:

"The TPM and AXLIB parameters" on page 841
An extended transactional client uses the TPM and AXLIB parameters to locate the transaction manager's ax_reg and ax_unreg functions. These functions are used only if the queue manager uses dynamic registration.

"Recovery following a failure in extended transactional processing" on page 841
Following a failure, a transaction manager must be able to recover any incomplete units of work. To do this, the transaction manager must be able to open as a resource manager any queue manager that was participating in an incomplete unit of work at the time of the failure.

The format of an xa_open string:

An xa_open string contains pairs of defined parameter names and values.

An xa_open string has the following format:

```
parm_name1=parm_value1,parm_name2=parm_value2, ...
```

where *parm_name* is the name of a parameter and *parm_value* is the value of a parameter. The names of the parameters are not case-sensitive but, unless stated otherwise, the values of the parameters are case-sensitive. You can specify the parameters in any order.

The names, meanings, and valid values of the parameters are as follows:

Name Meaning and valid values

CHANNEL
The name of an MQI channel.

This is an optional parameter. If this parameter is supplied, the CONNAME parameter must also be supplied.

TRPTYPE
The communications protocol for the MQI channel. The following are valid values:

LU62 SNA LU 6.2

NETBIOS
NetBIOS

SPX IPX/SPX

TCP TCP/IP

This is an optional parameter. If it is omitted, the default value of TCP is assumed. The values of the parameter are not case-sensitive.

CONNAME
The network address of the queue manager at the server end of the MQI channel. The valid values of this parameter depend on the value of the TRPTYPE parameter:

LU62 A symbolic destination name, which identifies a CPI-C side information entry.

The network qualified name of a partner LU is not a valid value, nor is a partner LU alias. This is because there are no additional parameters to specify a transaction program (TP) name and a mode name.

NETBIOS
A NetBIOS name.

SPX A 4 byte network address, a 6 byte node address, and an optional 2 byte socket number. These values must be specified in hexadecimal notation. A period must separate the network and node addresses, and the socket number, if supplied, must be enclosed in parentheses. For example:

```
0a0b0c0d.804abcde23a1(5e86)
```

If the socket number is omitted, the default value of 5e86 is assumed.

TCP A host name or an IP address, optionally followed by a port number in parentheses. If the port number is omitted, the default value of 1414 is assumed.

This is an optional parameter. If this parameter is supplied, the CHANNEL parameter must also be supplied.

QMNAME
The name of the queue manager at the server end of the MQI channel. The name cannot be blank or a single asterisk (*), nor can the name start with an asterisk. This means that the parameter must identify a specific queue manager by name.

This is a mandatory parameter.

When a client application is connected to a specific queue manager any transaction recovery must be processed by the same queue manager.

If the application is connecting to a z/OS queue manager then the application can specify either the name of a specific queue manager or the name of a queue-sharing group (QSG). By using the queue manager name or QSG name, the application controls whether it partakes in a transaction with a QMGR unit of recovery disposition or a GROUP unit of recovery disposition. The GROUP unit of recovery disposition enables the recovery of the transaction to be processed on any member of the QSG. To use GROUP units of recovery the **GROUPUR** queue manager attribute must be enabled.

TPM The transaction manager being used. The valid values are CICS and TUXEDO.

An extended transactional client uses this parameter and the AXLIB parameter for the same purpose. For more information these parameters, see The TPM and AXLIB parameters.

This is an optional parameter. The values of the parameter are not case-sensitive.

AXLIB
The name of the library that contains the transaction manager's ax_reg and ax_unreg functions.

This is an optional parameter.

Here is an example of an xa_open string:

```
channel=MARS.SVR,trptype=tcp,conname=MARS(1415),qmname=MARS,tpm=cics
```

The CHANNEL, TRPTYPE, CONNAME, and QMNAME parameters of the xa_open string:

Use this information to understand how the extended transactional client uses these parameters to determine the queue manager to connect to.

If the CHANNEL and CONNAME parameters are supplied in the xa_open string, the extended transactional client uses these parameters and the TRPTYPE parameter to start an MQI channel to the server queue manager.

If the CHANNEL and CONNAME parameters are not supplied in the xa_open string, the extended transactional client uses the value of the MQSERVER environment variable to start an MQI channel. If the MQSERVER environment variable is not defined, the extended transactional client uses the entry in the client channel definition identified by the QMNAME parameter.

In each of these cases, the extended transactional client checks that the value of the QMNAME parameter is the name of the queue manager at the server end of the MQI channel. If it is not, the xa_open call fails and the transaction manager reports the failure to the application.

If the applicat.ibmion client is connecting to a z/OS queue manager at V7.0.1 or later it can specify a queue-sharing group (QSG) name for the QMNAME parameter. This allows the application client to participate in a transaction with a GROUP unit of recovery disposition.

If the application uses a QSG name in QMNAME parameter field and the GROUPUR property is disabled on the queue manager to which it connects then the xa_open call fails.

If the application connects to a queue manager at an earlier version than V7.0.1, the xa_open call succeeds but the transaction has a QMGR unit of recovery disposition. Ensure that applications that require the GROUP unit of recovery disposition connect only to queue managers at V7.0.1 or later.

When the client application later calls MQCONN or MQCONNX on the same thread that the transaction manager used to issue the xa_open call, the application receives a connection handle for the MQI channel that was started by the xa_open call. A second MQI channel is not started. The extended transactional client checks that the value of the *QMgrName* parameter on the MQCONN or MQCONNX call is the name of the queue manager at the server end of the MQI channel. If it is not, the MQCONN or MQCONNX call fails with a reason code of MQRC_ANOTHER_Q_MGR_CONNECTED. If the value of the *QMgrName* parameter is blank or a single asterisk (*), or starts with an asterisk, the MQCONN or MQCONNX call fails with a reason code of MQRC_Q_MGR_NAME_ERROR.

If the client application has already started an MQI channel by calling MQCONN or MQCONNX before the transaction manager calls xa_open on the same thread, the transaction manager uses this MQI channel instead. A second MQI channel is not started. The extended transactional client checks that the value of the QMNAME parameter in the xa_open string is the name of the server queue manager. If it is not, the xa_open call fails.

If a client application starts an MQI channel first, the value of the *QMgrName* parameter on the MQCONN or MQCONNX call can be blank or a single asterisk (*), or it can start with an asterisk. Under these circumstances, however, you must ensure that the queue manager to which the application connects is the same as the queue manager that the transaction manager intends to open as a resource manager when it later calls xa_open on the same thread. You might encounter fewer problems, therefore, if the value of the *QMgrName* parameter identifies the queue manager explicitly by name.

The TPM and AXLIB parameters:

An extended transactional client uses the TPM and AXLIB parameters to locate the transaction manager's ax_reg and ax_unreg functions. These functions are used only if the queue manager uses dynamic registration.

If the TPM parameter is supplied in an xa_open string, but the AXLIB parameter is not supplied, the extended transactional client assumes a value for the AXLIB parameter based on the value of the TPM parameter. See Table 118 for the assumed values of the AXLIB parameter.

Table 118. Assumed values of the AXLIB parameter

Value of TPM	Platform	Assumed value of AXLIB
CICS	AIX	/usr/lpp/encina/lib/libEncServer.a(EncServer_shr.o)
CICS	HP-UX	/opt/encina/lib/libEncServer.sl
CICS	Solaris	/opt/encina/lib/libEncServer.so
CICS	Windows systems	libEncServer
Tuxedo	AIX	/usr/lpp/tuxedo/lib/libtux.a(libtux.so.60)
Tuxedo	HP-UX	/opt/tuxedo/lib/libtux.sl
Tuxedo	Solaris	/opt/tuxedo/lib/libtux.so.60
Tuxedo	Windows systems	libtux

If the AXLIB parameter is supplied in an xa_open string, the extended transactional client uses its value to override any assumed value based on the value of the TPM parameter. The AXLIB parameter can also be used for a transaction manager for which the TPM parameter does not have a specified value.

Additional error processing for xa_open:

The xa_open call fails in certain circumstances.

Topics in this section describe situations in which the xa_open call fails. It also fails if any of the following situations occur:

- There are errors in the xa_open string.
- There is insufficient information to start an MQI channel.
- There is a problem while trying to start an MQI channel (the server queue manager is not running, for example).

Recovery following a failure in extended transactional processing:

Following a failure, a transaction manager must be able to recover any incomplete units of work. To do this, the transaction manager must be able to open as a resource manager any queue manager that was participating in an incomplete unit of work at the time of the failure.

If you ever need to change any configuration information, therefore, you must ensure that all incomplete units of work have been resolved before making the changes. Alternatively, you must ensure that the configuration changes do not affect the ability of the transaction manager to open the queue managers it needs to open. The following are examples of such configuration changes:

- Changing the contents of an xa_open string
- Changing the value of the MQSERVER environment variable
- Changing entries in the client channel definition table (CCDT)
- Deleting a server connection channel definition

The XA switch structures:

Two XA switch structures are supplied with the extended transactional client on each platform.

These switch structures are:

MQRMIXASwitch
: This switch structure is used by a transaction manager when a queue manager, acting as a resource manager, is not using dynamic registration.

MQRMIXASwitchDynamic
: This switch structure is used by a transaction manager when a queue manager, acting as a resource manager, uses dynamic registration.

These switch structures are located in the libraries shown in Table 119.

Table 119. WebSphere MQ libraries containing the XA switch structures

Platform	Library containing the XA switch structures
AIX HP-UX Linux Solaris	*MQ_INSTALLATION_PATH*/lib/libmqcxa
Windows systems	*MQ_INSTALLATION_PATH*\bin\mqcxa.dll [1]
MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.	

The name of the WebSphere MQ resource manager in each switch structure is MQSeries_XA_RMI, but many queue managers can share the same switch structure.

Related concepts:

"Dynamic registration and extended transactional processing"
Using dynamic registration is a form of optimization because it can reduce the number of xa_ function calls issued by the transaction manager.

Dynamic registration and extended transactional processing:

Using dynamic registration is a form of optimization because it can reduce the number of xa_ function calls issued by the transaction manager.

If a queue manager does not use dynamic registration, a transaction manager involves the queue manager in every unit of work. The transaction manager does this by calling xa_start, xa_end, and xa_prepare, even if the queue manager has no resources that are updated within the unit of work.

If a queue manager uses dynamic registration, a transaction manager starts by assuming that the queue manager is not involved in a unit of work, and does not call xa_start. The queue manager then becomes involved in the unit of work only if its resources are updated within sync point control. If this occurs, the extended transactional client calls ax_reg to register the queue manager's involvement.

Using the extended transactional client with SSL channels:

You cannot set up an SSL channel using the xa_open string. Follow these instructions to use the client channel definition table (ccdt).

About this task

Because of the limited size of the xa_open xa_info string, it is not possible to pass all the information required to set up an SSL channel using the xa_open string method of connecting to a queue manager. Therefore you must either use the client channel definition table or, if your transaction manager allows, create the channel with MQCONNX before issuing the xa_open call.

To use the client channel definition table, follow these steps:

Procedure

1. Specify an xa_open string containing only the mandatory qmname (queue manager name) parameter, for example: `XA_Open_String=qmname=MYQM`
2. Use a queue manager to define a CLNTCONN (client-connection) channel with the required SSL parameters. Include the queue manager name in the QMNAME attribute on the CLNTCONN definition. This will be matched up with the qmname in the xa_open string.
3. Make the CLNTCONN definition available to the client system in a client channel definition table (CCDT) or, on Windows, in the active directory.
4. If you are using a CCDT, identify the CCDT containing the definition of the CLNTCONN channel using environment variables MQCHLLIB and MQCHLTAB. Set these variables in the environments used by both the client application and the transaction manager.

Results

This gives the transaction manager a channel definition to the appropriate queue manager with the SSL attributes needed to authenticate correctly, including SSLCIPH, the CipherSpec.

Configuring an extended transactional client for CICS:

You configure an extended transactional client for use by CICS by adding an XAD resource definition to a CICS region.

Add the XAD resource definition by using the CICS resource definition online (RDO) command, **`cicsadd`**. The XAD resource definition specifies the following information:

- An xa_open string
- The fully qualified path name of a switch load file

One switch load file is supplied for use by CICS on each of the following platforms: AIX, HP-UX, Solaris, and Windows systems. Each switch load file contains a function that returns a pointer to the XA switch structure that is used for dynamic registration, MQRMIXASwitchDynamic. See Table 120 on page 844 for the fully qualified path name of each switch load file.

Table 120. The switch load files

Platform	Switch load file
AIX HP-UX Linux Solaris	*MQ_INSTALLATION_PATH*/lib/amqczsc
Windows systems	*MQ_INSTALLATION_PATH*\bin\mqcc4swi.dll [1]
MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.	

Here is an example of an XAD resource definition for Windows systems:

```
cicsadd -c xad -r REGION1 WMQXA \
    ResourceDescription="WebSphere MQ queue manager MARS" \
    XAOpen="channel=MARS.SVR,trptype=tcp,conname=MARS(1415),qmname=MARS,tpm=cics" \
    SwitchLoadFile="C:\Program Files\IBM\WebSphere MQ\bin\mqcc4swi.dll"
```

For more information about adding an XAD resource definition to a CICS region, see the *CICS Administration Reference* and the *CICS Administration Guide* for your platform.

Note the following information about using CICS with an extended transactional client:

- You can add only one XAD resource definition for WebSphere MQ to a CICS region. This means that only one queue manager can be associated with a region, and all CICS applications that run in the region can connect only to that queue manager. If you want to run CICS applications that connect to a different queue manager, you must run the applications in a different region.
- Each application server in a region calls xa_open while it is initializing and starts an MQI channel to the queue manager associated with the region. This means that the queue manager must be started before an application server starts, otherwise the xa_open call fails. All WebSphere MQ MQI client applications later processed by the application server use the same MQI channel.
- When an MQI channel starts, and there is no security exit at the client end of the channel, the user ID that flows from the client system to the server connection MCA is `cics`. Under certain circumstances, the queue manager uses this user ID for authority checks when the server connection MCA subsequently attempts to access the queue manager resources on behalf of a client application. If this user ID is used for authority checks, you must ensure that it has the authority to access all the resources it needs to access.

  For information about when the queue manager uses this user ID for authority checks, see Security.
- The CICS task termination exits that are supplied for use on WebSphere MQ client systems are listed in Table 121. You configure these exits in the same way that you configure the corresponding exits for WebSphere MQ server systems. For this information, therefore, see the Enabling CICS user exits.

Table 121. CICS task termination exits

Platform	Source	Library
AIX HP-UX Linux Solaris	amqzscgx.c	amqczscg
Windows systems	amqzscgn.c	mqcc1415.dll

Configuring an extended transactional client for Tuxedo:

To configure XAD resource definition for use by Tuxedo, update the UBBCONFIG file and resource manager table.

To configure XAD resource definition for use by Tuxedo, perform the following actions:

- In the GROUPS section of the Tuxedo UBBCONFIG file for an application, use the OPENINFO parameter to specify an xa_open string.

  For an example of how to do this, see the sample UBBCONFIG file, which is supplied for use with the Tuxedo sample programs. On AIX, HP-UX, and Solaris, the name of the file is ubbstxcx.cfg and, on Windows systems, the name of the file is `ubbstxcn.cfg`.
- In the entry for a queue manager in the Tuxedo resource manager table:
  - udataobj/RM (AIX, HP-UX, and Solaris)
  - udataobj\rm (Windows systems)

  specify the name of an XA switch structure and the fully qualified path name of the library that contains the structure. For an example of how to do this for each platform, see TUXEDO samples. Tuxedo supports dynamic registration of a resource manager, and so you can use either MQRMIXASwitch or MQRMIXASwitchDynamic.

Microsoft Transaction Server

No additional configuration is required before you can use MTS as a transaction manager. However, there are some points to note.

Note the following information about using MTS with the extended transactional client:

- An MTS application always starts an MQI channel when it connects to a server queue manager. MTS, in its role as a transaction manager, then uses the same MQI channel to communicate with the queue manager.
- Following a failure, MTS must be able to recover any incomplete units of work. To do this, MTS must be able to communicate with any queue manager that was participating in an incomplete unit of work at the time of the failure.

  When an MTS application connects to a server queue manager and starts an MQI channel, the extended transactional client extracts sufficient information from the parameters of the MQCONN or MQCONNX call to enable the channel to be restarted following a failure, if required. The extended transactional client passes the information to MTS, and MTS records the information in its log.

  If the MTS application issues an MQCONN call, this information is simply the name of the queue manager. If the MTS application issues an MQCONNX call and provides a channel definition structure, MQCD, the information also includes the name of the MQI channel, the network address of the server queue manager, and the communications protocol for the channel.

  In a recovery situation, MTS passes this information back to the extended transactional client, and the extended transactional client uses it to restart the MQI channel.

  If you ever need to change any configuration information, therefore, ensure that all incomplete units of work have been resolved before making the changes. Alternatively, ensure that the configuration changes do not affect the ability of the extended transactional client to restart an MQI channel using the information recorded by MTS. The following are examples of such configuration changes:
  - Changing the value of the MQSERVER environment variable
  - Changing entries in the client channel definition table (CCDT)
  - Deleting a server connection channel definition
- Note the following conditions when using an extended transactional client with MTS:
  - Within a single thread, a client application can be connected to only one queue manager at a time.
  - Each thread of a client application can connect to a different queue manager.
  - A client application cannot use shared connection handles.

Defining MQI channels

To create a new channel, you have to create **two** channel definitions, one for each end of the connection, using the same channel name and compatible channel types. In this case, the channel types are *server-connection* and *client-connection*.

User defined channels

When the server does not automatically define channels there are two ways of creating the channel definitions and giving the WebSphere MQ application on the WebSphere MQ MQI client machine access to the channel.

These two methods are described in detail:

1. Create one channel definition on the WebSphere MQ client and the other on the server.

   This applies to any combination of WebSphere MQ MQI client and server platforms. Use it when you are getting started on the system, or to test your setup.

   See "Creating server-connection and client-connection definitions on different platforms" on page 847 for details on how to use this method.
2. Create both channel definitions on the server machine.

   Use this method when you are setting up multiple channels and WebSphere MQ MQI client machines at the same time.

   See "Creating server-connection and client-connection definitions on the server" on page 850 for details on how to use this method.

Automatically defined channels

WebSphere MQ products on platforms other than z/OS include a feature that can automatically create a channel definition on the server if one does not exist.

If an inbound attach request is received from a client and an appropriate server-connection definition cannot be found on that queue manager, WebSphere MQ creates a definition automatically and adds it to the queue manager. The automatic definition is based on the definition of the default server-connection channel SYSTEM.AUTO.SVRCONN. You enable automatic definition of server-connection definitions by updating the queue manager object using the ALTER QMGR command with the CHAD parameter (or the PCF command Change Queue Manager with the ChannelAutoDef parameter).

For more information about the automatic creation of channel definitions, see Auto-definition of receiver and server-connection channels.

Related concepts:

"Automatically defined channels"
WebSphere MQ products on platforms other than z/OS include a feature that can automatically create a channel definition on the server if one does not exist.

"User defined channels" on page 847
When the server does not automatically define channels there are two ways of creating the channel definitions and giving the WebSphere MQ application on the WebSphere MQ MQI client machine access to the channel.

"Channel control function" on page 787
The channel control function provides facilities for you to define, monitor, and control channels.

Automatically defined channels

WebSphere MQ products on platforms other than z/OS include a feature that can automatically create a channel definition on the server if one does not exist.

If an inbound attach request is received from a client and an appropriate server-connection definition cannot be found on that queue manager, WebSphere MQ creates a definition automatically and adds it to the queue manager. The automatic definition is based on the definition of the default server-connection channel SYSTEM.AUTO.SVRCONN. You enable automatic definition of server-connection definitions by updating the queue manager object using the ALTER QMGR command with the CHAD parameter (or the PCF command Change Queue Manager with the ChannelAutoDef parameter).

User defined channels

When the server does not automatically define channels there are two ways of creating the channel definitions and giving the WebSphere MQ application on the WebSphere MQ MQI client machine access to the channel.

These two methods are described in detail:

1. Create one channel definition on the WebSphere MQ client and the other on the server.

   This applies to any combination of WebSphere MQ MQI client and server platforms. Use it when you are getting started on the system, or to test your setup.

   See "Creating server-connection and client-connection definitions on different platforms" for details on how to use this method.
2. Create both channel definitions on the server machine.

   Use this method when you are setting up multiple channels and WebSphere MQ MQI client machines at the same time.

   See "Creating server-connection and client-connection definitions on the server" on page 850 for details on how to use this method.

Creating server-connection and client-connection definitions on different platforms

You can create each channel definition on the computer to which it applies. There are restrictions on how you can create channel definitions on a client computer.

On all platforms, you can use WebSphere MQ Script (MQSC) commands, programmable command format (PCF) commands, or the WebSphere MQ Explorer to define a server-connection channel on the server machine.

Because MQSC commands are not available on a machine where WebSphere MQ has been installed as a WebSphere MQ MQI client only, you must use different ways of defining a client-connection channel on the client machine.

Related concepts:

"Creating a client-connection channel on the WebSphere MQ MQI client" on page 848
You can define a client-connection channel on the client workstation using MQSERVER or using the MQCNO structure on an MQCONNX call.

Related tasks:

"Defining a server-connection channel on the server"
Start MQSC if necessary, then define the server-connection channel.

Defining a server-connection channel on the server

Start MQSC if necessary, then define the server-connection channel.

Procedure

1. Optional: If your server platform is not z/OS, first create and start a queue manager and then start MQSC commands.
   a. Create a queue manager, called QM1 for example:

      ```
      crtmqm QM1
      ```

b. Start the queue manager:

```
strmqm QM1
```

c. Start MQSC commands:

```
runmqsc QM1
```

2. Define a channel with your chosen name and a channel type of *server-connection*.

```
DEFINE CHANNEL(CHAN1) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
DESCR('Server-connection to Client_1')
```

   This channel definition is associated with the queue manager running on the server.

3. Use the following command to allow the inbound connect access to your queue manager:

```
SET CHLAUTH(CHAN1) TYPE(ADDRESSMAP)  ADDRESS('IP address') MCAUSER('userid')
```

   - Where SET CHLAUTH uses the name of the channel defined in the previous step.
   - Where *'IP address'* is the IP address of the client.
   - Where *'userid'* is the ID you want to provide to the channel for access control to the target queues. This field is case-sensitive.

   You can choose to identify your inbound connection using a number of different attributes. The example uses IP address. Alternative attributes include client user ID and SSL or TLS Subject Distinguished Name. For more information, see Channel authentication records

Creating a client-connection channel on the WebSphere MQ MQI client

You can define a client-connection channel on the client workstation using MQSERVER or using the MQCNO structure on an MQCONNX call.

Using MQSERVER

You can use the MQSERVER environment variable to specify a simple definition of a client-connection channel. It is simple in the sense that you can specify only a few attributes of the channel using this method.

- Specify a simple channel definition on Windows as follows:

```
SET MQSERVER=ChannelName/TransportType/ConnectionName
```

- Specify a simple channel definition on UNIX and Linux systems as follows:

```
export MQSERVER=ChannelName/TransportType/ConnectionName
```

- Specify a simple channel definition on IBM i systems as follows:

```
ADDENVVAR ENVVAR(MQSERVER) VALUE('ChannelName/TransportType/ConnectionName')
```

where:

- `ChannelName` must be the same name as defined on the server. It cannot contain a forward slash.
- `TransportType` can be one of the following values, depending on your IBM WebSphere MQ MQI client platform:
  - LU62
  - TCP
  - NETBIOS
  - SPX

  Note: On UNIX and Linux systems, the TransportType is case-sensitive and must be uppercase. An **MQCONN** or **MQCONNX** call returns 2058 if the TransportType is not recognized
- `ConnectionName` is the name of the server as defined to the communications protocol (TransportType).

For example, on Windows:

```
SET MQSERVER=CHANNEL1/TCP/MCID66499
```

or, on UNIX and Linux systems:

```
export MQSERVER=CHANNEL1/TCP/'MCID66499'
```

Note: To change the TCP/IP port number, see “MQSERVER” on page 874.

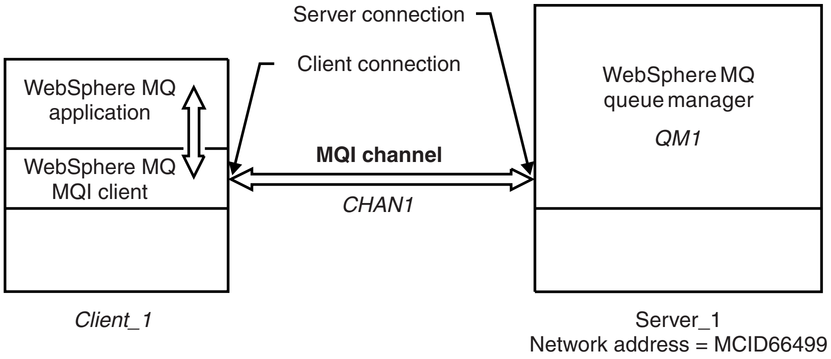


Figure 124. Simple channel definition

Some more examples of simple channel definitions are:

- On Windows:

```
SET MQSERVER=CHANNEL1/TCP/9.20.4.56
SET MQSERVER=CHANNEL1/NETBIOS/BOX643
```

- On UNIX and Linux systems:

```
export MQSERVER=CHANNEL1/TCP/'9.20.4.56'
export MQSERVER=CHANNEL1/LU62/BOX99
```

where `BOX99` is the LU 6.2 ConnectionName.

On the IBM WebSphere MQ MQI client, all **MQCONN** or **MQCONNX** requests then attempt to use the channel you have defined, unless the channel is overridden in an MQCD structure referenced from the MQCNO structure supplied to **MQCONNX**.

Note: For more information about the *MQSERVER* environment variable, see “MQSERVER” on page 874.

Using the MQCNO structure on an MQCONNX call

A IBM WebSphere MQ MQI client application can use the connect options structure, MQCNO, on an **MQCONNX** call to reference a channel definition structure, MQCD, that contains the definition of a client-connection channel.

In this way, the client application can specify the **ChannelName**, **TransportType**, and **ConnectionName** attributes of a channel at run time, enabling the client application to connect to multiple server queue managers simultaneously.

Note that if you define a channel using the *MQSERVER* environment variable, it is not possible to specify the **ChannelName**, **TransportType**, and **ConnectionName** attributes at run time.

A client application can also specify attributes of a channel such as **MaxMsgLength** and **SecurityExit**. Specifying such attributes enables the client application to specify values for the attributes that are not the default values, and enables channel exit programs to be called at the client end of an MQI channel.

If a channel uses the Secure Sockets Layer (SSL) or Transport Layer Security (TLS), a client application can also provide information relating to SSL or TLS in the MQCD structure. Additional information relating to SSL or TLS can be provided in the SSL or TLS configuration options structure, MQSCO, which is also referenced by the MQCNO structure on an **MQCONNX** call.

For more information about the MQCNO, MQCD, and MQSCO structures, see MQCNO, MQCD, and MQSCO.

Note: The sample program for MQCONNX is called **amqscnxc**. Another sample program called **amqssslc** demonstrates use of the MQSCO structure.

Creating server-connection and client-connection definitions on the server

You can create both definitions on the server, then make the client-connection definition available to the client.

First define a server-connection channel and then define a client-connection channel. On all platforms, you can use WebSphere MQ Script (MQSC) commands, programmable command format (PCF) commands or the WebSphere MQ Explorer to define a server-connection channel on the server machine.

Client-connection channel definitions created on the server are made available to clients using a client channel definition table (CCDT).

Related concepts:

"Client channel definition table"
The client channel definition table (CCDT) determines the channel definitions and authentication information used by client applications to connect to the queue manager. On platforms other than z/OS a CCDT is created automatically. You must then make it available to the client application.

Related tasks:

"Defining the server-connection channel on the server" on page 852
Create a server-connection channel definition for the queue manager.

"Defining the client-connection channel on the server" on page 853
Having defined the server-connection channel, you now define the corresponding client-connection channel.

"Accessing client-connection channel definitions" on page 854
Make the client channel definition table (CCDT) available to client applications by copying or sharing it, then specify its location and name on the client computer.

Client channel definition table

The client channel definition table (CCDT) determines the channel definitions and authentication information used by client applications to connect to the queue manager. On platforms other than z/OS a CCDT is created automatically. You must then make it available to the client application.

The purpose of the client channel definition table (CCDT) is to determine the channel definitions used by client applications to connect to the queue manager. The channel definition also specifies the authentication information that applies to the connections.

The CCDT is a binary file. It is generated by a queue manager. The queue manager does not read the CCDT file.

On platforms other than z/OS, the CCDT is created when the queue manager is created. Client connection channels are added to the table when you use the **DEFINE CHANNEL** command, and their definitions altered when you issue the **ALTER CHANNEL** command.

You can use the CCDT to provide clients with the authentication information to check for SSL certificate revocation. Define a namelist containing authentication information objects and set the queue manager attribute **SSLCRLNameList** to the name of the namelist.

There are a number of ways for a client application to use a CCDT. The CCDT can be copied to the client computer. You can copy the CCDT to a location shared by more than one client. You can make the CCDT accessible to the client as a shared file, while it remains located on the server.

If you use FTP to copy the file, use the `bin` option to set binary mode; do not use the default `ASCII` mode. Whichever method you choose to make the CCDT available, the location must be secure to prevent unauthorized changes to the channels.

Platforms other than z/OS

A default CCDT called `AMQCLCHL.TAB` is created when you create a queue manager.

By default, AMQCLCHL.TAB is located in the following directory on a server:

- UNIX Linux On UNIX and Linux systems:

  /*prefix*/qmgrs/*QUEUEMANAGERNAME*/@ipcc

  The name of the directory referenced by *QUEUEMANAGERNAME* is case-sensitive on UNIX and Linux systems. The directory name might not be the same as the queue manager name, if the queue manager name has special characters in it.
- Windows On Windows:

  MQ_INSTALLATION_PATH\data\qmgrs\QUEUEMANAGERNAME\@ipcc

MQ_INSTALLATION_PATH represents the high-level directory in which WebSphere MQ is installed.

However, you might have chosen to use a different directory for queue manager data. You can specify the parameter **-md** *DataPath* when you used the **crtmqm** command. If you do, `AMQCLCHL.TAB` is located in the `@ipcc` directory of the *DataPath* you specified.

The path to the CCDT can be changed by setting `MQCHLLIB`. If you do set `MQCHLLIB`, be aware, if you have multiple queue managers on the same server, they share the same CCDT location.

The CCDT is created when the queue manager is created. Each entry of a CCDT represents a client connection to a specific queue manager. A new entry is added when you define a client-connection channel using the **DEFINE CHANNEL** command, and the entry is updated when you alter the client-connection channels by using the **ALTER CHANNEL** command.

How to specify the location of the CCDT on the client

On a client system, you can specify the location of the CCDT in two ways:

- Using the environment variables `MQCHLLIB` to specify the directory where the table is located, and `MQCHLTAB` to specify the file name of the table.
- Using the client configuration file. In the `CHANNELS` stanza, use the attributes `ChannelDefinitionDirectory` to specify the directory where the table is located, and `ChannelDefinitionFile` to specify the file name.

If the location is specified both in the client configuration file and by using environment variables, the environment variables take priority. You can use this feature to specify a standard location in the client configuration file and override it using environment variables when necessary.

Related reference:

"MQCHLLIB" on page 871
MQCHLLIB specifies the directory path to the file containing the client channel definition table (CCDT). The file is created on the server, but can be copied across to the WebSphere MQ MQI client workstation.

Related information:

Working with revoked certificates

Migration and client channel definition tables (CCDT):

In general, the internal format of the client channel definition table might change from one release level of IBM WebSphere MQ to the next. As a result, a IBM WebSphere MQ MQI client can use a client channel definition table only when it has been prepared by a server queue manager that is at the same release level as the client, or at an earlier release level.

A Version 7.1 IBM WebSphere MQ MQI client can use a client channel definition table that has been prepared by a Version 6.0 queue manager. But a Version 6.0 client cannot use a client channel definition table that has been prepared by a Version 7.1 queue manager.

Client connection channels in the Active Directory

On Windows systems that support the Active Directory, IBM WebSphere MQ publishes client connection channels in the Active Directory to provide dynamic client-server binding.

When client connection channel objects are defined, they are written into a client channel definition file, called AMQCLCHL.TAB by default. If the client connection channels use the TCP/IP protocol, the IBM WebSphere MQ server also publishes them in the Active Directory. When the IBM WebSphere MQ client determines how to connect to the server, it looks for a relevant client connection channel object definition using the following search order:

1. MQCONNX MQCD data structure
2. MQSERVER environment variable
3. client channel definition file
4. Active Directory

This order means that any current applications are not affected by any change. You can think of these entries in the Active Directory as records in the client channel definition file, and the IBM WebSphere MQ client processes them in the same way. To configure and administer support for publishing client connection channel definitions in the Active Directory, use the **setmqscp** command, as described in setmqscp.

Defining the server-connection channel on the server

Create a server-connection channel definition for the queue manager.

Procedure

1. On the server machine, define a channel with your chosen name and a channel type of *server-connection*. For example:

   ```
   DEFINE CHANNEL(CHAN2) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
   DESCR('Server-connection to Client_2')
   ```

2. Use the following command to allow the inbound connect access to your queue manager:

   ```
   SET CHLAUTH(CHAN2) TYPE(ADDRESSMAP)  ADDRESS('IP address') MCAUSER('userid')
   ```

   - Where SET CHLAUTH uses the name of the channel defined in the previous step.
   - Where *'IP address'* IP address is the IP address of the client.
   - Where *'userid'* is the ID you want to provide to the channel for access control to the target queues. This field is case-sensitive.

You can choose to identify your inbound connection using a number of different attributes. The example uses IP address. Alternative attributes include client user ID and SSL or TLS Subject Distinguished Name. For more information, see Channel authentication records This channel definition is associated with the queue manager running on the server.

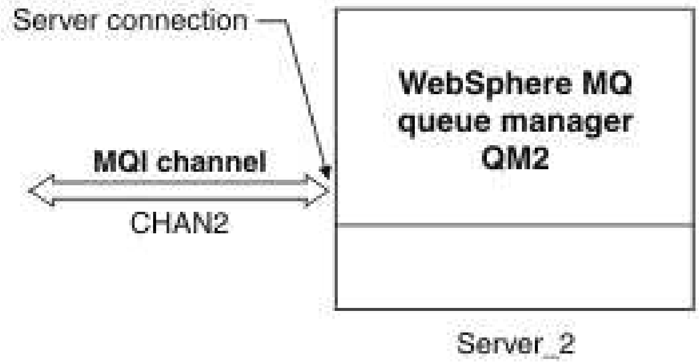


Figure 125. Defining the server-connection channel

Defining the client-connection channel on the server

Having defined the server-connection channel, you now define the corresponding client-connection channel.

Before you begin

Define the server-connection channel.

Procedure

1. Define a channel with the same name as the server-connection channel, but a channel type of *client-connection*. You must state the connection name (CONNAME). For TCP/IP, the connection name is the network address or host name of the server machine. It is also advisable to specify the queue manager name (QMNAME) to which you want your IBM WebSphere MQ application, running in the client environment, to connect. By varying the queue manager name, you can define a set of channels to connect to different queue managers.

   ```
   DEFINE CHANNEL(CHAN2) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
   CONNAME(9.20.4.26) QMNAME(QM2) DESCR('Client-connection to Server_2')
   ```

2. Use the following command to allow the inbound connect access to your queue manager:

   ```
   SET CHLAUTH(CHAN2) TYPE(ADDRESSMAP)  ADDRESS('IP-address') MCAUSER('userid')
   ```

   - Where SET CHLAUTH uses the name of the channel defined in the previous step.
   - Where *'IP address'* is the IP address of the client.
   - Where *'userid'* is the ID you want to provide to the channel for access control to the target queues. This field is case-sensitive.

   You can choose to identify your inbound connection using a number of different attributes. The example uses IP address. Alternative attributes include client user ID and SSL or TLS Subject Distinguished Name. For more information, see Channel authentication records

Results

On platforms other than z/OS, this channel definition is stored in a file called the client channel definition table (CCDT), which is associated with the queue manager. The client channel definition table can contain more than one client-connection channel definition. For more information about the client channel definition table, and for the corresponding information about how client-connection channel definitions are stored on z/OS, see “Client channel definition table” on page 850.

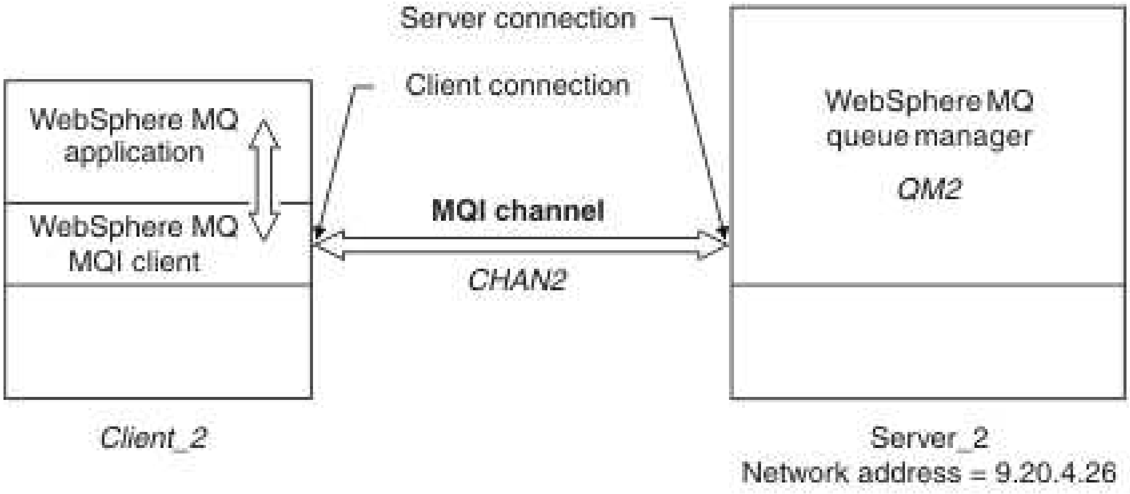


Figure 126. Defining the client-connection channel

Accessing client-connection channel definitions

Make the client channel definition table (CCDT) available to client applications by copying or sharing it, then specify its location and name on the client computer.

Before you begin

You have defined the client-connection channels you need.

On z/OS, you have created a CCDT. On other platforms, the CCDT is automatically created and updated.

About this task

For a client application to use the client channel definition table (CCDT), you must make the CCDT available to it and specify its location and name

Procedure

1. Make the CCDT available to the client applications in one of three ways:
   a. Optional: Copy the CCDT to the client computer.
   b. Optional: Copy the CCDT to a location shared by more than one client.
   c. Optional: Leave the CCDT on the server but make it shareable by the client.

   Whichever location you choose for the CCDT, the location must be secure to prevent unauthorized changes to the channels.
2. On the client, specify the location and name of the file containing the CCDT in one of three ways:
   a. Optional: Use the CHANNELS stanza of the client configuration file. For more information, see "CHANNELS stanza of the client configuration file" on page 862.
   b. Optional: Use the environment variables MQCHLLIB and MQCHLTAB.

      For example, you can set the environment variables by typing:
      - On HP Integrity NonStop Server, and UNIX and Linux systems:

        ```
        export MQCHLLIB=MQ_INSTALLATION_PATH/qmgrs/QUEUEMANAGERNAME/@ipcc
        export MQCHLTAB=AMQCLCHL.TAB
        ```

      where *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed.

c. Optional: On Windows only, use the **setmqscp** control command to publish the client-connection channel definitions in Active Directory

If the MQSERVER environment variable is set, a WebSphere MQ client uses the client-connection channel definition specified by MQSERVER in preference to any definitions in the client channel definition table.

Channel-exit programs for MQI channels

Three types of channel exit are available to the WebSphere MQ MQI client environment on UNIX, Linux and Windows systems.

These are:

- Send exit
- Receive exit
- Security exit

These exits are available at both the client and the server end of the channel. Exits are not available to your application if you are using the MQSERVER environment variable. Channel exits are explained in Channel exit programs for messaging channels.

The send and receive exits work together. There are several possible ways in which you can use them:

- Splitting and reassembling a message
- Compressing and decompressing data in a message (this functionality is provided as part of WebSphere MQ, but you might want to use a different compression technique)
- Encrypting and decrypting user data (this functionality is provided as part of WebSphere MQ, but you might want to use a different encryption technique)
- Journaling each message sent and received

You can use the security exit to ensure that the WebSphere MQ client and server are correctly identified, and to control access.

If send or receive exits on the server-connection side of the channel instance need to perform MQI calls on the connection with which they are associated, they use the connection handle provided in the `MQCXP Hconn` field. You must be aware that client-connection send and receive exits cannot make MQI calls.

Related concepts:

"Security exits on a client connection" on page 856
You can use security exit programs to verify that the partner at the other end of a channel is genuine. Special considerations apply when a security exit is applied to a client connection.

Related reference:

"Path to exits" on page 856
A default path for location of the channel exits is defined in the client configuration file. Channel exits are loaded when a channel is initialized.

"Identifying the API call in a send or receive exit program" on page 857
When you use MQI channels for clients, byte 10 of the agent buffer identifies the API call in use when a send or receive exit is called. This is useful for identifying which channel flows include user data and might require processing such as encryption or digital signing.

Related information:

Extending queue manager facilities

User exits, API exits, and WebSphere MQ installable services

Path to exits

A default path for location of the channel exits is defined in the client configuration file. Channel exits are loaded when a channel is initialized.

On UNIX, Linux and Windows systems, a client configuration file is added to your system during installation of the WebSphere MQ MQI client. A default path for location of the channel exits on the client is defined in this file, using the stanza:

```
ClientExitPath:
  ExitsDefaultPath=string
  ExitsDefaultPath64=string
```

where *string* is a file location in a format appropriate to the platform

When a channel is initialized, after an **MQCONN** or **MQCONNX** call, the client configuration file is searched. The ClientExitPath stanza is read and any channel exits that are specified in the channel definition are loaded.

Security exits on a client connection

You can use security exit programs to verify that the partner at the other end of a channel is genuine. Special considerations apply when a security exit is applied to a client connection.

Figure 127 on page 857 illustrates the use of security exits in a client connection, using the WebSphere MQ object authority manager to authenticate a user. Either SecurityParmsPtr or SecurityParmsOffset is set in the MQCNO structure on the client and there are security exits at both ends of the channel. After the normal security message exchange has ended, and the channel is ready to run, the MQCSP structure accessed from the MQCXP SecurityParms field is passed to the security exit on the client. The exit type is set to MQXR_SEC_PARMS. The security exit can elect to do nothing to the user identifier and password, or it can alter either or both of them. The data returned from the exit is then sent to the server-connection end of the channel. The MQCSP structure is rebuilt on the server-connection end of the channel and is passed to the server-connection security exit accessed from the MQCXP SecurityParms field. The security exit receives and processes this data. This processing is typically to reverse any change made to the user ID and password fields in the client exit, which are then used to authorize the queue manager connection. The resulting MQCSP structure is referenced using SecurityParmsPtr in the MQCNO structure on the queue manager system.

If SecurityParmsPtr or SecurityParmsOffset are set in the MQCNO structure and there is a security exit at only one end of the channel, the security exit receives and processes the MQCSP structure. Actions such as encryption are inappropriate for a single user exit, as there is no exit to perform the complementary action.

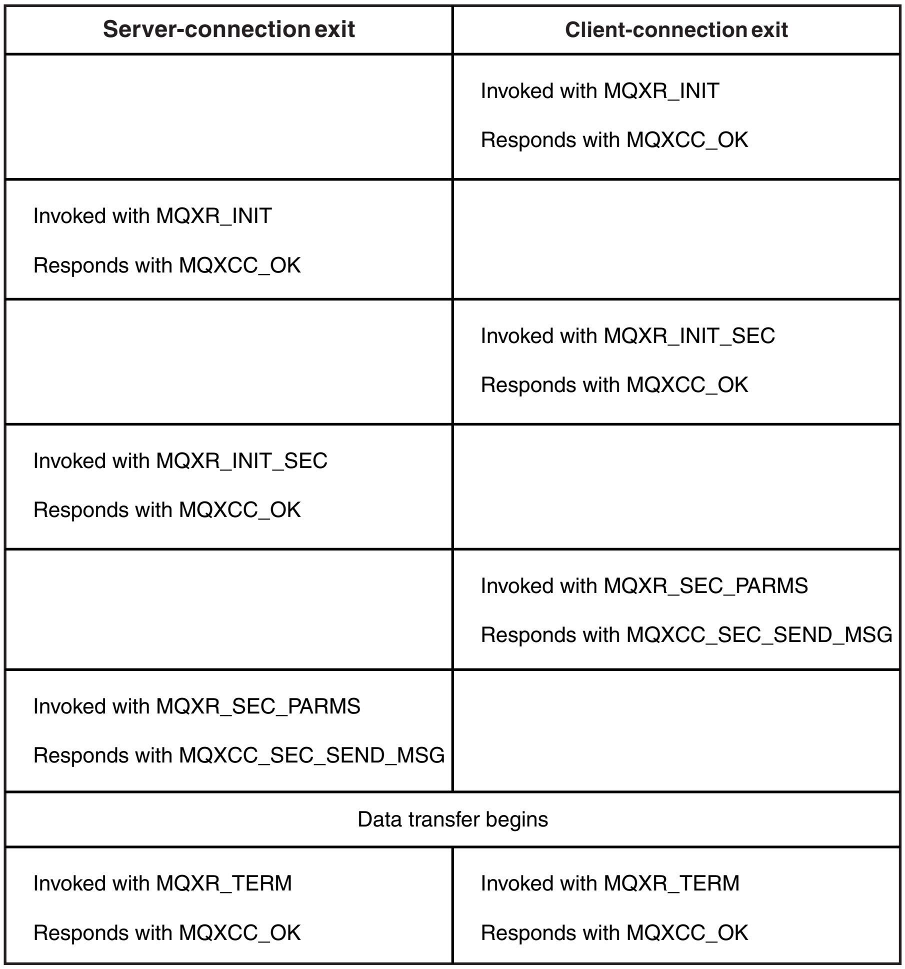
If SecurityParmsPtr and SecurityParmsOffset are not set in the MQCNO structure and there is a security exit at either or both ends of the channel, the security exit or exits are called. Either security exit can return its own MQCSP structure, addressed through the SecurityParmsPtr; the security exit is not called again until it is terminated (ExitReason of MQXR_TERM). The exit writer can free the memory used for the MQCSP at that stage.

When a server-connection channel instance is sharing more than one conversation, the pattern of calls to the security exit is restricted on the second and subsequent conversations.

For the first conversation, the pattern is the same as if the channel instance is not sharing conversations. For the second and subsequent conversations, the security exit is never called with MQXR_INIT, MQXR_INIT_SEC, or MQXR_SEC_MSG. It is called with MQXR_SEC_PARMS.

In a channel instance with sharing conversations, MQXR_TERM is called only for the last conversation running.

Each conversation has the opportunity in the MQXR_SEC_PARMS invocation of the exit to alter the MQCD; on the server-connection end of the channel this feature can be useful to vary, for example, the MCAUserIdentifier or LongMCAUserIdPtr values before the connection is made to the queue manager.



Server-connection exit	Client-connection exit
	Invoked with MQXR_INIT Responds with MQXCC_OK
Invoked with MQXR_INIT Responds with MQXCC_OK	
	Invoked with MQXR_INIT_SEC Responds with MQXCC_OK
Invoked with MQXR_INIT_SEC Responds with MQXCC_OK	
	Invoked with MQXR_SEC_PARMS Responds with MQXCC_SEC_SEND_MSG
Invoked with MQXR_SEC_PARMS Responds with MQXCC_SEC_SEND_MSG	
Data transfer begins	
Invoked with MQXR_TERM Responds with MQXCC_OK	Invoked with MQXR_TERM Responds with MQXCC_OK

Figure 127. Client connection-initiated exchange with agreement for client connection using security parameters

Note: Security exit applications constructed prior to the release of WebSphere MQ v7.1 may require updating. For more information see Channel security exit programs.

Identifying the API call in a send or receive exit program

When you use MQI channels for clients, byte 10 of the agent buffer identifies the API call in use when a send or receive exit is called. This is useful for identifying which channel flows include user data and might require processing such as encryption or digital signing.

The following table shows the data that appears in byte 10 of the channel flow when an API call is being processed.

Note: These are not the only values of this byte. There are other **reserved** values.

Table 122. Identifying API calls

API call	**Value of byte 10 for request**	**Value of byte 10 for reply**
MQCONN [1,2]	X'81'	X'91'
MQDISC [1]	X'82'	X'92'
MQOPEN [3]	X'83'	X'93'
MQCLOSE	X'84'	X'94'
MQGET [4]	X'85'	X'95'
MQPUT [4]	X'86'	X'96'
MQPUT1 request [4]	X'87'	X'97'
MQSET request	X'88'	X'98'
MQINQ request	X'89'	X'99'
MQCMIT request	X'8A'	X'9A'
MQBACK request	X'8B'	X'9B'
MQSTAT request	X'8D'	X'9D'
MQSUB request	X'8E'	X'9E'
MQSUBRQ request	X'8F'	X'9F'
xa_start request	X'A1'	X'B1'
xa_end request	X'A2'	X'B2'
xa_open request	X'A3'	X'B3'
xa_close request	X'A4'	X'B4'
xa_prepare request	X'A5'	X'B5'
xa_commit request	X'A6'	X'B6'
xa_rollback request	X'A7'	X'B7'
xa_forget request	X'A8'	X'B8'
xa_recover request	X'A9'	X'B9'
xa_complete request	X'AA'	X'BA'

Notes:

1. The connection between the client and server is initiated by the client application using MQCONN. Therefore, for this command in particular, there are several other network flows. The same applies to MQDISC, which terminates the network connection.
2. MQCONNX is treated in the same way as MQCONN for the purposes of the client-server connection.
3. If a large distribution list is opened, there might be more than one network flow per MQOPEN call in order to pass all the required data to the SVRCONN MCA.
4. Large messages can exceed the transmission segment size. If this happens there can be many network flows resulting from a single API call.

Connecting a client to a queue-sharing group

You can connect a client to a queue-sharing group by creating an MQI channel between a client and a queue manager on a server that is a member of a queue-sharing group.

A queue-sharing group is formed by a set of queue-managers that can access the same set of shared queues.

A client putting to a shared queue can connect to any member of the queue-sharing group. The benefits of connecting to a queue-sharing group are possible increases in front-end and back-end availability, and increased capacity. You can connect to a specific queue manager or to the generic interface.

Connecting directly to a queue manager in a queue-sharing group gives the benefit that you can put messages to a shared target queue, which increases back-end availability.

Connecting to the generic interface of a queue-sharing group opens a session with one of the queue managers in the group. This increases front-end availability, because the client queue manager can connect with any queue-manager in the group. You connect to the group using the generic interface when you do not want to connect to a specific queue manager within the queue-sharing group.

The generic interface can be a WLM/DNS group name or a VTAM® generic resource name, or another common interface to the queue-sharing group.

To connect to the generic interface of a queue-sharing group you need to create channel definitions that can be accessed by any queue manager in the group. To do this you need to have the same definitions on each queue manager in the group.

Define the SVRCONN channel as follows:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
MCAUSER(' ') QSGDISP(GROUP)
```

Channel definitions on the server are stored in a shared DB2 repository. Each queue manager in the queue-sharing group makes a local copy of the definition, ensuring that you will always connect to the correct server-connection channel when you issue an MQCONN or MQCONNX call.

Define the CLNTCONN channel as follows:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME(WLM/DNS groupname) QMNAME(QSG1) +
DESCR('Client-connection to Queue Sharing Group QSG1') QSGDISP(GROUP)
```

Because the generic interface of the queue-sharing group is stored in the CONNAME field in the client-connection channel, you can now connect to any queue manager in the group, and put to shared queues owned by that group.

Related concepts:

"Creating channel definitions"
To connect to the generic interface of a queue-sharing group you need to create channel definitions that can be accessed by any queue manager in the group. To do this you need to have the same definitions on each queue manager in the group.

Creating channel definitions

To connect to the generic interface of a queue-sharing group you need to create channel definitions that can be accessed by any queue manager in the group. To do this you need to have the same definitions on each queue manager in the group.

Define the SVRCONN channel as follows:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
MCAUSER(' ') QSGDISP(GROUP)
```

Channel definitions on the server are stored in a shared DB2 repository. Each queue manager in the queue-sharing group makes a local copy of the definition, ensuring that you will always connect to the correct server-connection channel when you issue an MQCONN or MQCONNX call.

Define the CLNTCONN channel as follows:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME(WLM/DNS groupname) QMNAME(QSG1) +
DESCR('Client-connection to Queue Sharing Group QSG1') QSGDISP(GROUP)
```

Because the generic interface of the queue-sharing group is stored in the CONNAME field in the client-connection channel, you can now connect to any queue manager in the group, and put to shared queues owned by that group.

Configuring a client using a configuration file

Configure your clients using attributes in a text file. These attributes can be overridden by environment variables or in other platform-specific ways.

You configure your WebSphere MQ MQI clients using a text file, similar to the queue manager configuration file, qm.ini, used on UNIX and Linux platforms. The file contains a number of stanzas, each of which contains a number of lines of the format **attribute-name**=*value*.

In this documentation, this file is referred to as the *WebSphere MQ MQI client configuration file*; its file name is generally mqclient.ini, but you can choose to give it another name. Configuration information in this file applies to all platforms, and to clients using the MQI, WebSphere MQ classes for Java, WebSphere MQ classes for JMS, WebSphere MQ classes for .NET, and XMS.

The configuration features apply to all connections a client application makes to any queue managers, rather than being specific to an individual connection to a queue manager. Attributes relating to a connection to an individual queue manager can be configured programmatically, for example by using an MQCD structure, or by using a Client Channel Definition Table (CCDT).

Environment variables which were supported in releases of WebSphere MQ earlier than Version 7.0 continue to be supported, and where such an environment variable matches an equivalent value in the client configuration file, the environment variable overrides the client configuration file value.

For a client application using WebSphere MQ classes for JMS, you can also override the client configuration file in the following ways:

- setting properties in the JMS configuration file
- setting Java system properties, which also overrides the JMS configuration file

For the .NET client, you can also override the client configuration file and the equivalent environment variables using the .NET application configuration file.

Note that you cannot set up multiple channel connections using the client configuration file.

Example client configuration file

```
#* Module Name: mqclient.ini                                  *#
#* Type       : WebSphere MQ MQI client configuration file        *#
#  Function   : Define the configuration of a client          *#
#*                                                            *#
#*************************************************************#
#* Notes      :                                               *#
#* 1) This file defines the configuration of a client         *#
#*                                                                  *#
#********************************************************************#

ClientExitPath:
   ExitsDefaultPath=/var/mqm/exits
   ExitsDefaultPath64=/var/mqm/exits64

TCP:
   Library1=DLLName1
   KeepAlive = Yes
   ClntSndBuffSize=32768
   ClntRcvBuffSize=32768
   Connect_Timeout=0

MessageBuffer:
   MaximumSize=-1
   Updatepercentage=-1
   PurgeTime=0

LU62:
   TPName
   Library1=DLLName1
   Library2=DLLName2

PreConnect:
 Module=amqldapi
 Function=myFunc
 Data=ldap://myLDAPServer.com:389/cn=wmq,ou=ibm,ou=com
 Sequence=1

CHANNELS:
 DefRecon=YES
 ServerConnectionParms=SALES.SVRCONN/TCP/hostname.x.com(1414)
```

Related reference:

"Location of the client configuration file"
A IBM WebSphere MQ MQI client configuration file can be held in a number of locations.

"CHANNELS stanza of the client configuration file" on page 862
Use the CHANNELS stanza to specify information about client channels.

"ClientExitPath stanza of the client configuration file" on page 864
Use the ClientExitPath stanza to specify the default locations of channel exits on the client.

"LU62, NETBIOS, and SPX stanzas of the client configuration file" on page 865
On Windows systems only, use these stanzas to specify configuration parameters for the specified network protocols.

"MessageBuffer stanza of the client configuration file" on page 866
Use the MessageBuffer stanza to specify information about message buffers.

"SSL stanza of the client configuration file" on page 867
Use the SSL stanza to specify information about the use of SSL or TLS.

"TCP stanza of the client configuration file" on page 869
Use the TCP stanza to specify TCP network protocol configuration parameters.

"Using WebSphere MQ environment variables" on page 870
This section describes the environment variables that you can use with WebSphere MQ MQI client applications.

"Changing queue manager configuration information" on page 1157
The attributes described here modify the configuration of an individual queue manager. They override any settings for WebSphere MQ.

Location of the client configuration file

A IBM WebSphere MQ MQI client configuration file can be held in a number of locations.

A client application uses the following search path to locate the IBM WebSphere MQ MQI client configuration file:

1. The location specified by the environment variable MQCLNTCF.

   The format of this environment variable is a full URL. This means the file name might not necessarily be `mqclient.ini` and facilitates placing the file on a network attached file-system.

   Note the following:

   - C, .NET and XMS clients support only the `file:` protocol; the `file:` protocol is assumed if the URL string does not begin with `protocol:`
   - To allow for Java 1.4.2 JREs, which do not support reading environment variables, the MQCLNTCF environment variable can be overridden with an MQCLNTCF Java System Property.
2. A file called `mqclient.ini` in the present working directory of the application.
3. A file called `mqclient.ini` in the IBM WebSphere MQ data directory for Windows, UNIX and Linux systems.

   Note the following:

   - The IBM WebSphere MQ data directory does not exist on certain platforms, for example, IBM i and z/OS, or in cases where the client has been supplied with another product.
   - On UNIX and Linux systems, the directory is `/var/mqm`
   - On Windows platforms you configure the environment variable MQ_FILE_PATH during installation, to point at the data directory. It is normally `C:\Program Files\IBM\WebSphere MQ`
   - To allow for Java 1.4.2 JREs that do not support reading environment variables you can manually override the MQ_FILE_PATH environment variable with an MQ_FILE_PATH Java System Property.
4. A file called `mqclient.ini` in a standard directory appropriate to the platform, and accessible to users:
   - For all Java clients this is the value of the `user.home` Java System Property.
   - For C clients on UNIX and Linux platforms this is the value of the HOME environment variable.
   - For C clients on Windows this is the concatenated values of the HOMEDRIVE and HOMEPATH environment variables.

Note: For the IBM WebSphere MQ client for HP Integrity NonStop Server, the `mqclient.ini` file must be located in the OSS file system. Guardian applications must either place the `mqclient.ini` file in the IBM WebSphere MQ data directory or set the MQCLNTCF environment variable to a location in the OSS file system.

CHANNELS stanza of the client configuration file

Use the CHANNELS stanza to specify information about client channels.

The following attributes can be included in the CHANNELS stanza:

CCSID=*number*

The coded character set number to be used.

The CCSID number is equivalent to the MQCCSID environment parameter.

ChannelDefinitionDirectory=*path*

The directory path to the file containing the client channel definition table.

On Windows systems, the default is the IBM WebSphere MQ installation directory, typically `C:\Program Files\IBM\WebSphere MQ`. On UNIX and Linux systems, the default is `/var/mqm`.

The ChannelDefinitionDirectory path is equivalent to the MQCHLLIB environment parameter.

ChannelDefinitionFile=*filename*| AMQCLCHL.TAB

The name of the file containing the client channel definition table.

The client channel definition table is equivalent to the MQCHLTAB environment parameter.

ReconDelay=*(delay[,rand])(delay[,rand])...*

The ReconDelay attribute provides an administrative option to configure reconnect delay for client programs that can auto-reconnect. Here is an example configuration:

```
ReconDelay=(1000,200)(2000,200)(4000,1000)
```

The example shown defines an initial delay of one second, plus a random interval of up to 200 milliseconds. The next delay is two seconds plus a random interval of up to 200 milliseconds. All subsequent delays are four seconds, plus a random interval of up to 1000 milliseconds.

DefRecon=*NO*|*YES*|*QMGR*|*DISABLED*

The DefRecon attribute provides an administrative option to enable client programs to automatically reconnect, or to disable the automatic reconnection of a client program that has been written to reconnect automatically. You might opt to set the latter if a program uses an option, such as MQPMO_LOGICAL_ORDER, that is incompatible with reconnection.

The interpretation of the DefRecon options depends on whether an MQCNO_RECONNECT_* value is also set in the client program, and what value is set.

If the client program connects using MQCONN, or sets the MQCNO_RECONNECT_AS_DEF option using MQCONNX, the reconnect value set by DefRecon takes effect. If no reconnect value is set in the program, or by the DefRecon option, the client program is not reconnected automatically. Automatic client reconnection is not supported by IBM WebSphere MQ classes for Java.

NO Unless overridden by MQCONNX, the client is not reconnected automatically.

YES Unless overridden by MQCONNX, the client reconnects automatically.

QMGR Unless overridden by MQCONNX, the client reconnects automatically, but only to the same queue manager. The QMGR option has the same effect as MQCNO_RECONNECT_Q_MGR.

DISABLED

Reconnection is disabled, even if requested by the client program using the MQCONNX MQI call.

The automatic client reconnection depends on two values:

- The reconnect option set in the application
- DefRecon value in the mqclient.ini file

Table 123. Automatic reconnection depends on the values set in the application and in the mqclient.ini file

DefRecon value in the mqclient.ini	**Reconnection options set in the application**			
	MQCNO_RECONNECT	MQCNO_RECONNECT_Q_MGR	MQCNO_RECONNECT_AS_DEF	MQCNO_RECONNECT_DISABLED
NO	YES	QMGR	NO	NO
YES	YES	QMGR	YES	NO
QMGR	YES	QMGR	QMGR	NO
DISABLED	NO	NO	NO	NO

MQReconnectTimeout

The timeout in seconds to reconnect to a client. The default value is 1800 seconds (30 minutes).

ServerConnectionParms

ServerConnectionParms is equivalent to the MQSERVER environment parameter and specifies the location of the IBM WebSphere MQ server and the communication method to be used. The ServerConnectionParms attribute defines only a simple channel; you cannot use it to define an SSL channel or a channel with channel exits. It is a string of the format *ChannelName/TransportType/*

ConnectionName, *ConnectionName* must be a fully qualified network name. *ChannelName* cannot contain the forward slash ("/") character because this character is used to separate the channel name, transport type, and connection name.

When ServerConnectionParms is used to define a client channel, a maximum message length of 100 MB is used. Therefore the maximum message size in effect for the channel is the value specified in the SVRCONN channel on the server.
Note that only a single client channel connection can be made. For example, if you have two entries:

```
ServerConnectionParms=R1.SVRCONN/TCP/localhost(1963)
ServerConnectionParms=R2.SVRCONN/TCP/localhost(1863)
```

only the second one is used.
Specify `ConnectionName` as a comma-separated list of names for the stated transport type. Generally, only one name is required. You can provide multiple *hostnames* to configure multiple connections with the same properties. The connections are tried in the order that they are specified in the connection list until a connection is successfully established. If no connection is successful, the client starts to process again. Connection lists are an alternative to queue manager groups to configure connections for reconnectable clients.

`Put1DefaultAlwaysSync=`*`NO|YES`*
Controls the behavior of the MQPUT1 function call with the option `MQPMO_RESPONSE_AS_Q_DEF`.

NO If MQPUT1 is set with `MQPMO_SYNCPOINT`, it behaves as `MQPMO_ASYNC_RESPONSE`. Similarly, if MQPUT1 is set with `MQPMO_NO_SYNCPOINT`, it behaves as `MQPMO_SYNC_RESPONSE`. This is the default value.

YES MQPUT1 behaves as if `MQPMO_SYNC_RESPONSE` is set, regardless of whether `MQPMO_SYNCPOINT` or `MQPMO_NO_SYNCPOINT` is set.

ClientExitPath stanza of the client configuration file

Use the ClientExitPath stanza to specify the default locations of channel exits on the client.

The following attributes can be included in the ClientExitPath stanza:

`ExitsDefaultPath=`*`string`*
Specifies the location of 32-bit channel exits for clients.

`ExitsDefaultPath64=`*`string`*
Specifies the location of 64-bit channel exits for clients.

`JavaExitsClassPath=`*`string`*
The values to be added to the classpath when a Java exit is run. This is ignored by exits in any other language.

In the JMS configuration file, the JavaExitsClassPath name is given the standard com.ibm.mq.cfg. prefix and this full name is also used on the Websphere MQ V7.0 system property. At Version 6.0 this attribute was specified using system property com.ibm.mq.exitClasspath, which was documented in the Version 6.0 readme. The use of com.ibm.mq.exitClasspath is deprecated. If both JavaExitsClassPath and exitClasspath are present, JavaExitsClassPath is honored. If only exitClasspath usage is present, it is still honored in Websphere MQ V7.0.

LU62, NETBIOS, and SPX stanzas of the client configuration file

On Windows systems only, use these stanzas to specify configuration parameters for the specified network protocols.

LU62

Use the LU62 stanza to specify SNA LU 6.2 protocol configuration parameters. The following attributes can be included in this stanza:

Library1=*DLLName***|WCPIC32**
The name of the APPC DLL.

Library2=*DLLName***|WCPIC32**
The same as Library1, used if the code is stored in two separate libraries. .

TPName
The TP name to start on the remote site.

NETBIOS

Use the NETBIOS stanza to specify NetBIOS protocol configuration parameters. The following attributes can be included in this stanza:

AdapterNum=*number***|0**
The number of the LAN adapter.

Library1=*DLLName***|NETAPI32**
The name of the NetBIOS DLL.

LocalName=*name*
The name by which this computer is known on the LAN.

This is equivalent to the MQNAME environment parameter.

NumCmds=*number***|1**
How many commands to allocate.

NumSess=*number***|1**
How many sessions to allocate.

SPX

Use the SPX stanza to specify SPX protocol configuration parameters. The following attributes can be included in this stanza:

BoardNum=*number***|0**
The LAN adapter number.

KeepAlive=YES|NO
Switch the KeepAlive function on or off.

KeepAlive=YES causes SPX to check periodically that the other end of the connection is still available. If it is not, the channel is closed.

Library1=*DLLName***|WSOCK32.DLL**
The name of the SPX DLL.

Library2=*DLLName***|WSOCK32.DLL**
The same as Library1, used if the code is stored in two separate libraries.

Socket=*number***|5E86**
The SPX socket number in hexadecimal notation.

MessageBuffer stanza of the client configuration file

Use the MessageBuffer stanza to specify information about message buffers.

The following attributes can be included in the MessageBuffer stanza:

`MaximumSize=`*`integer`***`|1`**
: Size, in kilobytes, of the read-ahead buffer, in the range 1 through 999 999.

  The following special values exist:

  -1 The client determines the appropriate value.

  0 Read ahead is disabled for the client.

`PurgeTime=`*`integer`***`|600`**
: Interval, in seconds, after which messages left in the read-ahead buffer are purged.

  If the client application is selecting messages based on MsgId or CorrelId it is possible that the read ahead buffer might contain messages sent to the client with a previously requested MsgId or CorrelId. These messages would then be stranded in the read ahead buffer until an MQGET is issued with an appropriate MsgId or CorrelId. You can purge messages from the read ahead buffer by setting PurgeTime. Any messages that have remained in the read ahead buffer for longer than the purge interval are automatically purged. These messages have already been removed from the queue on the queue manager, so unless they are being browsed, they are lost.

  The valid range is in the range 1 through 999 999 seconds, or the special value 0, meaning that no purge takes place.

`UpdatePercentage=`*`integer`***`|-1`**
: The update percentage value, in the range of 1 - 100, used in calculating the threshold value to determine when a client application makes a new request to the server. The special value -1 indicates that the client determines the appropriate value.

  The client periodically sends a request to the server indicating how much data the client application has consumed. A request is sent when the number of bytes, *n*, retrieved by the client by way of MQGET calls exceeds a threshold *T*. *n* is reset to zero each time a new request is sent to the server.

  The threshold T is calculated as follows:

  ```
  T = Upper - Lower
  ```

  Upper is the same as the read-ahead buffer size, specified by the *MaximumSize* attribute, in Kilobytes. Its default is 100 Kb.

  Lower is lower than Upper, and is specified by the *UpdatePercentage* attribute. This attribute is a number in the range 1 through 100, and has a default of 20. Lower is calculated as follows:

  ```
  Lower = Upper x UpdatePercentage / 100
  ```

  Example 1:

  The MaximumSize and UpdatePercentage attributes take their defaults of 100 Kb and 20 Kb.

  The client calls MQGET to retrieve a message, and does so repeatedly. This continues until MQGET has consumed n bytes.

  Using the calculation

  ```
  T = Upper - Lower
  ```

  T is (100 - 20) = 80 Kb.

  So when MQGET calls have removed 80 Kb from a queue, the client makes a new request automatically.

  Example 2:

  The MaximumSize attributes takes its default of 100 Kb, and a value of 40 is chosen for UpdatePercentage.

The client calls MQGET to retrieve a message, and does so repeatedly. This continues until MQGET has consumed n bytes.

Using the calculation

```
T = Upper - Lower
```

T is (100 - 40) = 60 Kb

So when MQGET calls have removed 60 Kb from a queue, the client makes a new request automatically. This is sooner than in EXAMPLE 1 where the defaults were used.

Therefore choosing a larger threshold *T* tends to decrease the frequency at which requests are sent from client to server. Conversely choosing a smaller threshold *T* tends to increase the frequency of requests that are sent from client to server.

However, choosing a large threshold *T* can mean that the performance gain of read ahead is reduced as the chance of the read ahead buffer becoming empty can increase. When this happens an MQGET call might have to pause, waiting for data to arrive from the server.

SSL stanza of the client configuration file

Use the SSL stanza to specify information about the use of SSL or TLS.

The following attributes can be included in the SSL stanza:

CertificateValPolicy=*string*
Determines the type of certificate validation used.

ANY Use any certificate validation policy supported by the underlying secure sockets library. This setting is the default setting.

RFC5280
Use only certificate validation which complies with the RFC 5280 standard.

EncryptionPolicySuiteB=*string*
Determines whether a channel uses Suite-B compliant cryptography and what level of strength is to be used. The possible values are:

NONE
Suite-B compliant cryptography is not used. This setting is the default setting.

128_BIT,192_BIT
Sets the security strength to both 128-bit and 192-bit levels.

128_BIT
Sets the security strength to 128-bit level.

192_BIT
Sets the security strength to 192-bit level.

OCSPAuthentication=OPTIONAL|REQUIRED|WARN
Defines the behavior of WebSphere MQ when OCSP is enabled and the OCSP revocation check is unable to determine the certificate revocation status. There are three possible values:

OPTIONAL
Any certificate with a revocation status that cannot be determined by OCSP checking is accepted and no warning or error message is generated. The SSL or TLS connection continues as if no revocation check had been made.

REQUIRED
OCSP checking must yield a definitive revocation result for every SSL or TLS certificate which is checked. Any SSL or TLS certificate with a revocation status that cannot be verified is rejected with an error message. If queue manager SSL event messages are enabled, an

MQRC_CHANNEL_SSL_ERROR message with a ReasonQualifier of MQRQ_SSL_HANDSHAKE_ERROR is generated. The connection is closed.

This value is the default value.

WARN

A warning is reported in the queue manager error logs if an OCSP revocation check is unable to determine the revocation status of any SSL or TLS certificate. If queue manager SSL event messages are enabled, an MQRC_CHANNEL_SSL_WARNING message with a ReasonQualifier of MQRQ_SSL_UNKNOWN_REVOCATION is generated. The connection is allowed to continue

OCSPCheckExtensions=YES|NO

Controls whether WebSphere MQ acts on AuthorityInfoAccess certificate extensions. If the value is set to `NO`, WebSphere MQ ignores AuthorityInfoAccess certificate extensions and does not attempt an OCSP security check. The default value is `YES`.

SSLCryptoHardware=*string*

Sets the parameter string required to configure PKCS #11 cryptographic hardware present on the system.

Specify a string in the following format: **GSK_PKCS11**=*driver path and filename*;*token label*;*token password*;*symmetric cipher setting*;

For example: `GSK_PKCS11=/usr/lib/pkcs11/PKCS11_API.so;tokenlabel;passw0rd;SYMMETRIC_CIPHER_ON`

The driver path is an absolute path to the shared library providing support for the PKCS #11 card. The driver file name is the name of the shared library. An example of the value required for the PKCS #11 driver path and file name is `/usr/lib/pkcs11/PKCS11_API.so`. To access symmetric cipher operations through GSKit, specify the symmetric cipher setting parameter. The value of this parameter is either:

SYMMETRIC_CIPHER_OFF

Do not access symmetric cipher operations. This setting is the default setting.

SYMMETRIC_CIPHER_ON

Access symmetric cipher operations.

The maximum length of the string is 256 characters. The default value is blank. If you specify a string that is not in the correct format, an error is generated.

SSLFipsRequired=YES|NO

Specifies whether only FIPS-certified algorithms are to be used if cryptography is carried out in WebSphere MQ. If cryptographic hardware is configured, the cryptographic modules used are those modules provided by the hardware product. These might, or might not, be FIPS-certified to a particular level, depending on the hardware product in use.

SSLHTTPProxyName=*string*

The string is either the host name or network address of the HTTP Proxy server that is to be used by GSKit for OCSP checks. This address can be followed by an optional port number, enclosed in parentheses. If you do not specify the port number, the default HTTP port, 80, is used. On the HP-UX PA-RISC and Sun Solaris SPARC platforms, and for 32-bit clients on AIX, the network address can be only an IPv4 address; on other platforms it can be an IPv4 or IPv6 address.

This attribute might be necessary if, for example, a firewall prevents access to the URL of the OCSP responder.

SSLKeyRepository=*pathname*

The location of the key repository that holds the user's digital certificate, in stem format. That is, it includes the full path and the file name without an extension.

SSLKeyResetCount=*integer***|0**

The number of unencrypted bytes sent and received on an SSL or TLS channel before the secret key is renegotiated.

The value must be in the range 0 - 999999999.

The default is 0, which means that secret keys are never renegotiated.

If you specify a value of 1 - 32768, SSL or TLS channels use a secret key reset count of 32768 (32Kb). This is to avoid excessive key resets, which would occur for small secret key reset values.

TCP stanza of the client configuration file

Use the TCP stanza to specify TCP network protocol configuration parameters.

The following attributes can be included in the TCP stanza:

ClntRcvBuffSize=*number***| 32768**
The size in bytes of the TCP/IP receive buffer used by the client end of a client-connection server-connection channel. A value of zero indicates that the operating system will manage the buffer sizes, as opposed to the buffer sizes being fixed by WebSphere MQ.

ClntSndBuffSize=*number***| 32768**
The size in bytes of the TCP/IP send buffer used by the client end of a client-connection server-connection channel. A value of zero indicates that the operating system will manage the buffer sizes, as opposed to the buffer sizes being fixed by WebSphere MQ.

Connect_Timeout=*number*
The number of seconds before an attempt to connect the socket times out; the default is 0 unless the channel has been configured with a non-zero client channel weighting in which case the default is 5.

IPAddressVersion=MQIPADDR_IPV4|MQIPADDR_IPV6
Specifies which IP protocol to use for a channel connection.

It has the possible string values of MQIPADDR_IPV4 or MQIPADDR_IPV6. These values have the same meanings as IPV4 and IPV6 in **ALTER QMGR IPADDRV**.

KeepAlive=YES|NO
Switch the KeepAlive function on or off. KeepAlive=YES causes TCP/IP to check periodically that the other end of the connection is still available. If it is not, the channel is closed.

Library1=*DLLName***| WSOCK32**
(Windows only) The name of the TCP/IP sockets DLL.

TMF and TMF/Gateway stanzas

The IBM WebSphere MQ provided TMF/Gateway runs in a Pathway environment. Use the TMF and TMF/Gateway stanzas to specify the required configuration parameters for the IBM WebSphere MQ client for HP Integrity NonStop Server to communicate with the TMF/Gateway.

If you want to use TMF, then you must define a TMF stanza and one TmfGateway stanza for each queue manager with which you are communicating. All values are derived from your configuration.

TMF

PathMon=*name*
The name of your defined Pathmon process that defines the server classes for the TMF/Gateway.

TmfGateway

The following attributes can be included in this stanza:

QManager=*name*
The name of the queue manager.

Server=*name*
The server class name for the TMF/Gateway configured for that queue manager.

Example

Here is an example of a TMF stanza that is defined with two TmfGateway stanzas for two different queue managers on different servers:

```
TMF:
   PathMon=$PSD1P

TmfGateway:
   QManager=MQ5B
   Server=MQ-MQ5B

TmfGateway:
   QManager=MQ5C
   Server=MQ-MQ5C
```

Using WebSphere MQ environment variables

This section describes the environment variables that you can use with WebSphere MQ MQI client applications.

You can use environment variables in the following ways:

- Set the variables in your system profile to make a permanent change
- Issue a command from the command line to make a change for this session only
- To give one or more variables a particular value dependent on the application that is running, add commands to a command script file used by the application

WebSphere MQ uses default values for those variables that you have not set.

Commands are available on all the WebSphere MQ MQI client platforms unless otherwise stated.

For each environment variable, use the command relevant to your platform to display the current setting or to reset the value of a variable.

For example:

Setting or resetting the value of an environment variable

Effect	**Command**	
	Windows	**UNIX and Linux systems**
Removes the variable	SET MQSERVER=	unset MQSERVER
Displays the current setting	SET MQSERVER	echo $MQSERVER
Displays all environment variables for the session	set	set

For information about the individual variables, see the following subtopics:

Related concepts:

"Configuring a client using a configuration file" on page 860
Configure your clients using attributes in a text file. These attributes can be overridden by environment variables or in other platform-specific ways.

Related information:

Environment variables

MQCCSID

MQCCSID specifies the coded character set number to be used and overrides the CCSID value with which the server has been configured.

See Choosing client or server coded character set identifier (CCSID) for more information.

To set this variable use one of these commands:

- For Windows:
  ```
  SET MQCCSID=number
  ```
- For UNIX and Linux systems:
  ```
  export MQCCSID=number
  ```

MQCERTVPOL

MQCERTVPOL specifies the certificate validation policy used.

For more information about certificate validation policies in WebSphere MQ, see Certificate validation policies in WebSphere MQ.

This environment variable overrides the *CertificateValPolicy* setting in the SSL stanza of the client ini file. The variable can be set to one of two values:

ANY Use any certificate validation policy supported by the underlying secure sockets library.

RFC5280
Use only certificate validation which complies with the RFC 5280 standard.

To set this variable, use one of these commands:

- For Windows:
  ```
  SET MQCERTVPOL=value
  ```
- For UNIX and Linux systems:
  ```
  export MQCERTVPOL=value
  ```

MQCHLLIB

`MQCHLLIB` specifies the directory path to the file containing the client channel definition table (CCDT). The file is created on the server, but can be copied across to the WebSphere MQ MQI client workstation.

If `MQCHLLIB` is not set, the path for the client defaults to:

- Windows For Windows: *`MQ_INSTALLATION_PATH`*
- UNIX Linux For UNIX and Linux systems: `/var/mqm/`

For the **`crtmqm`** and **`strmqm`** commands, the path defaults to one of two sets of paths. If *`datapath`* is set, the path defaults to one of the first set. If *`datapath`* is not set, the path defaults to one of the second set.

- Windows For Windows: *`datapath`*`\@ipcc`
- UNIX Linux For UNIX and Linux systems: *`datapath`*`/@ipcc`

Or:

- Windows For Windows: *`MQ_INSTALLATION_PATH`*`\data\qmgrs\`*`qmgrname`*`\@ipcc`
- UNIX Linux For UNIX and Linux systems: `/prefix/qmgrs/`*`qmgrname`*`/@ipcc`

where:

- *`MQ_INSTALLATION_PATH`* represents the high-level directory in which WebSphere MQ is installed.
- If present, *`datapath`* is the value of `DataPath` defined in the queue manager stanza.

- *prefix* is the value of Prefix defined in the queue manager stanza. Prefix is typically /var/mqm on UNIX and Linux platforms.
- *qmgrname* is the value of the Directory attribute defined in the queue manager stanza. The value might be different from the actual queue manager name. The value might have been altered to replace special characters.
- The queue manager stanza is defined in the mqs.ini file on UNIX, and Linux, and in the registry on Windows

Notes:

1. If set, MQCHLLIB overrides the path used to locate the CCDT.
2. Environment variables, such as MQCHLLIB, can be scoped to a process, or a job, or system-wide, in a platform-specific way.
3. If you set MQCHLLIB system-wide on a server, it sets the same path to the CCDT file for all the queue managers on the server. If you do not set the MQCHLLIB environment variable, the path is different for each queue manager. Queue managers read the value of MQCHLLIB, if it is set, on either the **crtmqm** or **strmqm** command.
4. If you create multiple queue managers on one server, the distinction is important, for the following reason. If you set MQCHLLIB system-wide, each queue manager updates the same CCDT file. The file contains the client-connection definitions from all the queue managers on the server. If the same definition exists on multiple queue managers, SYSTEM.DEF.CLNTCONN for example, the file contains the latest definition. When you create a queue manager, if MQCHLLIB is set, SYSTEM.DEF.CLNTCONN is updated in the CCDT. The update overwrites the SYSTEM.DEF.CLNTCONN created by a different queue manager. If you modified the earlier definition, your modifications are lost. For this reason, you must consider finding alternatives to setting MQCHLLIB as a system-wide environment variable on the server.
5. The MQSC and PCF NOREPLACE option on a client-connection definition does not check the contents of the CCDT file. A client-connection channel definition of the same name that was previously created, but not by this queue manager, is replaced, regardless of the NOREPLACE option. If the definition was previously created by the same queue manager, the definition is not replaced.
6. The command, **rcrmqobj** -t clchltab deletes and recreates the CCDT file. The file is recreated with only the client-connection definitions created on the queue manager that the command is running against.
7. Other commands that update the CCDT modify only the client-connection channels that have the same channel name. Other client-connection channels in the file are not altered.
8. The path for MQCHLLIB does not need quotations marks.

Examples

To set this variable use one of these commands:

- Windows For Windows:

  ```
  SET MQCHLLIB=pathname
  ```

  For example:

  ```
  SET MQCHLLIB=C:\wmqtest
  ```

- UNIX Linux For UNIX and Linux systems:

  ```
  export MQCHLLIB=pathname
  ```

MQCHLTAB

MQCHLTAB specifies the name of the file containing the client channel definition table (ccdt). The default file name is AMQCLCHL.TAB.

For information about where the client channel definition table is located on a server, see “Client channel definition table” on page 850.

To set this variable use one of these commands:

- On Windows:

  ```
  SET MQCHLTAB=filename
  ```

- On UNIX and Linux systems:

  ```
  export MQCHLTAB=filename
  ```

For example:

```
SET MQCHLTAB=ccdf1.tab
```

In the same way as for the client, the MQCHLTAB environment variable on the server specifies the name of the client channel definition table.

MQIPADDRV

MQIPADDRV specifies which IP protocol to use for a channel connection. It has the possible string values of "MQIPADDR_IPV4" or "MQIPADDR_IPV6". These values have the same meanings as IPV4 and IPV6 in ALTER QMGR IPADDRV. If it is not set, "MQIPADDR_IPV4" is assumed.

To set this variable use one of these commands:

- For Windows:

  ```
  SET MQIPADDRV=MQIPADDR_IPV4|MQIPADDR_IPV6
  ```

- For UNIX and Linux systems:

  ```
  export MQIPADDRV=MQIPADDR_IPV4|MQIPADDR_IPV6
  ```

MQNAME

MQNAME specifies the local NetBIOS name that the WebSphere MQ processes can use.

See “Defining a NetBIOS connection on Windows” on page 821 for a full description and for the rules of precedence on the client and the server.

To set this variable use this command:

```
SET MQNAME=Your_env_Name
```

For example:

```
SET MQNAME=CLIENT1
```

The NetBIOS on some platforms requires a different name (set by MQNAME) for each application if you are running multiple WebSphere MQ applications simultaneously on the WebSphere MQ MQI client.

MQSERVER

MQSERVER environment variable is used to define a minimal channel. MQSERVER specifies the location of the WebSphere MQ server and the communication method to be used.

You cannot use MQSERVER to define an SSL channel or a channel with channel exits. For details of how to define an SSL channel, see Protecting channels with SSL.

ConnectionName must be a fully-qualified network name. The *ChannelName* cannot contain the forward slash (/) character because this character is used to separate the channel name, transport type, and connection name. When the MQSERVER environment variable is used to define a client channel, a maximum message length (MAXMSGL) of 100 MB is used. Therefore the maximum message size in effect for the channel is the value specified in the SVRCONN channel at the server.

To set this variable use one of these commands:

- For Windows:
  ```
  SET MQSERVER=ChannelName/TransportType/ConnectionName
  ```
- For UNIX and Linux systems:
  ```
  export MQSERVER='ChannelName/TransportType/ConnectionName'
  ```

TransportType can be one of the following values, depending on your IBM WebSphere MQ client platform:

- LU62
- TCP
- NETBIOS
- SPX

ConnectionName can be a comma-separated list of connection names. The connection names in the list are used in a similar way to multiple connections in a client connection table. The *ConnectionName* list might be used as an alternative to queue manager groups to specify multiple connections for the client to try. If you are configuring a multi-instance queue manager, you might use a *ConnectionName* list to specify different queue manager instances.

TCP/IP default port:

By default, for TCP/IP, WebSphere MQ assumes that the channel will be connected to port 1414.

You can change this by:

- Adding the port number in brackets as the last part of the ConnectionName:
  - For Windows:
    ```
    SET MQSERVER=ChannelName/TransportType/ConnectionName(PortNumber)
    ```
  - For UNIX and Linux systems:
    ```
    export MQSERVER='ChannelName/TransportType/ConnectionName(PortNumber)'
    ```
- Changing the `mqclient.ini` file by adding the port number to the protocol name, for example:
  ```
  TCP:
  port=2001
  ```
- Adding WebSphere MQ to the services file as described in “Using the TCP/IP listener” on page 828.

SPX default socket:

By default, for SPX, WebSphere MQ assumes that the channel will be connected to socket 5E86.

You can change this by:

- Adding the socket number in brackets as the last part of the ConnectionName:

  ```
  SET MQSERVER=ChannelName/TransportType/ConnectionName(SocketNumber)
  ```

  For SPX connections, specify the ConnectionName and socket in the form `network.node(socket)`. If the WebSphere MQ client and server are on the same network, the network need not be specified. If you are using the default socket, the socket need not be specified.
- Changing the `qm.ini` file by adding the port number to the protocol name, for example:

  ```
  SPX:
  socket=5E87
  ```

Using MQSERVER:

If you use the MQSERVER environment variable to define the channel between your WebSphere MQ MQI client machine and a server machine, this is the only channel available to your application, and no reference is made to the client channel definition table (CCDT).

In this situation, the listener program that you have running on the server machine determines the queue manager to which your application will connect. It will be the same queue manager as the listener program is connected to.

If the MQCONN or MQCONNX request specifies a queue manager other than the one the listener is connected to, or if the MQSERVER parameter *TransportType* is not recognized, the MQCONN or MQCONNX request fails with return code MQRC_Q_MGR_NAME_ERROR.

OnUNIX and Linux systems, you might define MQSERVER as in one of the following examples:

```
export MQSERVER=CHANNEL1/TCP/'9.20.4.56(2002)'
export MQSERVER=CHANNEL1/LU62/BOX99
```

All **MQCONN** or **MQCONNX** requests then attempt to use the channel you have defined unless an MQCD structure has been referenced from the MQCNO structure supplied to **MQCONNX**, in which case the channel specified by the MQCD structure takes priority over any specified by the MQSERVER environment variable.

The MQSERVER environment variable takes priority over any client channel definition pointed to by MQCHLLIB and MQCHLTAB.

Canceling MQSERVER

To cancel MQSERVER and return to the client channel definition table pointed to by MQCHLLIB and MQCHLTAB, enter the following:

- On Windows:

  ```
  SET MQSERVER=
  ```
- On UNIX and Linux systems:

  ```
  unset MQSERVER
  ```

MQSNOAUT

When you set this variable it disables the object authority manager (OAM) and prevents any security checking.

The MQSNOAUT variable only takes effect when a queue manager is created.

MQSSLCRYP

MQSSLCRYP holds a parameter string that allows you to configure the cryptographic hardware present on the system. The permitted values are the same as for the SSLCRYP parameter of the ALTER QMGR command.

To set this variable use one of these commands:

- On Windows systems:
```
SET MQSSLCRYP=string
```
- On UNIX and Linux systems:
```
export MQSSLCRYP=string
```

MQSSLFIPS

MQSSLFIPS specifies whether only FIPS-certified algorithms are to be used if cryptography is carried out in WebSphere MQ. The values are the same as for the SSLFIPS parameter of the ALTER QMGR command.

The use of FIPS-certified algorithms is affected by the use of cryptographic hardware, see Specifying that only FIPS-certified CipherSpecs are used at run time on the MQI client.

To set this variable use one of these commands:

- On Windows systems:
```
SET MQSSLFIPS=YES|NO
```
- On UNIX and Linux systems:
```
export MQSSLFIPS=YES|NO
```

The default is `NO`.

MQSSLKEYR

MQSSLKEYR specifies the location of the key repository that holds the digital certificate belonging to the user, in stem format. Stem format means that it includes the full path and the file name without an extension. For full details, see the SSLKEYR parameter of the ALTER QMGR command.

To set this variable use one of these commands:

- On Windows systems:
```
SET MQSSLKEYR=pathname
```
- On UNIX and Linux systems:
```
export MQSSLKEYR=pathname
```

There is no default value.

MQSSLPROXY

MQSSLPROXY specifies the host name and port number of the HTTP proxy server to be used by GSKit for OCSP checks.

To set this variable use one of these commands:

- On Windows systems:

```
SET MQSSLPROXY=string
```

- On UNIX and Linux systems:

```
export MQSSLPROXY="string"
```

The string is either the host name or network address of the HTTP Proxy server which is to be used by GSKit for OCSP checks. This address can be followed by an optional port number, enclosed in parentheses. If you do not specify the port number, the default HTTP port, 80, is used.

For example, on UNIX and Linux systems, you can use the one of the following commands:

- `export MQSSLPROXY="proxy.example.com(80)"`
- `export MQSSLPROXY="127.0.0.1"`

MQSSLRESET

MQSSLRESET represents the number of unencrypted bytes sent and received on an SSL or TLS channel before the secret key is renegotiated.

See Resetting SSL and TLS secret keys for more information about secret key renegotiation.

It can be set to an integer in the range 0 through 999 999 999. The default is 0, which indicates that secret keys are never renegotiated. If you specify an SSL or TLS secret key reset count in the range 1 byte through 32 KB, SSL or TLS channels use a secret key reset count of 32 KB. This secret reset count is to avoid excessive key resets which would occur for small SSL or TLS secret key reset values.

To set this variable use one of these commands:

- On Windows systems:

```
SET MQSSLRESET=integer
```

- On UNIX and Linux systems:

```
export MQSSLRESET=integer
```

Controlling queued publish/subscribe

You can start, stop and display the status of queued publish/subscribe. You can also add and remove streams, and add and delete queue managers from a broker hierarchy.

See the following subtopics for more information on controlling queued publish/subscribe:

Setting queued publish/subscribe message attributes

You control the behavior of some publish/subscribe message attributes using queue manager attributes. The other attributes you control in the *Broker* stanza of the `qm.ini` file.

About this task

You can set the following publish/subscribe attributes: for details see, Queue manager parameters

Table 124. Publish/subscribe configuration parameters

Description	MQSC parameter name
Command message retry count	**PSRTYCNT**
Discard undeliverable command input message	**PSNPMSG**
Behavior following undeliverable command response message	**PSNPRES**
Process command messages under syncpoint	**PSSYNCPT**

The Broker stanza is used to manage the following configuration settings:

- `PersistentPublishRetry=yes | force`

  If you specify `Yes`, then if a publication of a persistent message through the queued publish/subscribe interface fails, and no negative reply was requested, the publish operation is retried.

  If you requested a negative response message, the negative response is sent and no further retry occurs.

  If you specify `Force`, then if a publication of a persistent message through the queued publish/subscribe interface fails, the publish operation is retried until the it is successfully processed. No negative response is sent.

- `NonPersistentPublishRetry=yes | force`

  If you specify `Yes`, then if a publication of a non-persistent message through the queued publish/subscribe interface fails, and no negative reply was requested, the publish operation is retried.

  If you requested a negative response message, the negative response is sent and no further retry occurs.

  If you specified `Force`, then if a publication of a non-persistent message through the queued publish/subscribe interface fails, the publish operation is retried until it is successfully processed. No negative response is sent.

  Note: If you want to enable this functionality for non-persistent messages, then as well as setting the `NonPersistentPublishRetry` value you must also ensure that the queue manager attribute **PSSYNCPT** is set to `Yes`.

  Doing this might also have an impact on the performance of processing non-persistent publications as the **MQGET** from the STREAM queue now occurs under syncpoint.

- `PublishBatchSize=`*number*

  The broker normally processes publish messages within syncpoint. It can be inefficient to commit each publication individually, and in some circumstances the broker can process multiple publish messages in a single unit of work. This parameter specifies the maximum number of publish messages that can be processed in a single unit of work

  The default value for `PublishBatchSize` is 5.

- `PublishBatchInterval=`*number*

  The broker normally processes publish messages within syncpoint. It can be inefficient to commit each publication individually, and in some circumstances the broker can process multiple publish messages in a single unit of work. This parameter specifies the maximum time (in milliseconds) between the first message in a batch and any subsequent publication included in the same batch.

  A batch interval of 0 indicates that up to `PublishBatchSize` messages can be processed, provided that the messages are available immediately.

  The default value for `PublishBatchInterval` is zero.

Procedure

Use WebSphere MQ Explorer, programmable commands, or the **runmqsc** command to alter the queue manager attributes that control the behavior of publish/subscribe.

Example

ALTER QMGR PSNPRES(SAFE)

Starting queued publish/subscribe

Before you begin

Read the description of PSMODE to understand the three modes of publish/subscribe:

- COMPAT
- DISABLED
- ENABLED

Note: If you have migrated from Version 6.0 you must use **strmqbrk** to migrate Version 6.0 publish/subscribe broker state if you are working with an upgraded queue manager. This does not apply to z/OS.

About this task

Set the QMGR PSMODE attribute to start either the queued publish/subscribe interface (also known as the broker), or the publish/subscribe engine (also known as Version 7 publish/subscribe) or both. To start queued publish/subscribe you need to set PSMODE to ENABLED. The default is ENABLED.

Procedure

Use WebSphere MQ Explorer or the **runmqsc** command to enable the queued publish/subscribe interface if the interface is not already enabled.

Example

```
ALTER QMGR PSMODE(ENABLED)
```

What to do next

WebSphere MQ processes queued publish/subscribe commands and publish/subscribe Message Queue Interface (MQI) calls.

Stopping queued publish/subscribe

Before you begin

Queued publish/subscribe is deprecated.

Read the description of PSMODE to understand the three modes of publish/subscribe:

- COMPAT
- DISABLED
- ENABLED

About this task

Set the QMGR PSMODE attribute to stop either the queued publish/subscribe interface (also known as the broker), or the publish/subscribe engine (also known as Version 7 publish/subscribe) or both. To stop queued publish/subscribe you need to set PSMODE to COMPAT. To stop the publish/subscribe engine entirely, set PSMODE to DISABLED.

Procedure

Use WebSphere MQ Explorer or the **runmqsc** command to disable the queued publish/subscribe interface.

Example

```
ALTER QMGR PSMODE(COMPAT)
```

Adding a stream

You can add streams manually so that they coexist with streams migrated from Version 6.0 queue managers.

Before you begin

Familiarize yourself with the way publish/subscribe streams operate by reading the topic, Streams and topics.

About this task

Use PCF command, **runmqsc**, or IBM WebSphere MQ Explorer to do these steps.

Note: You can perform steps 1 and 2 in any order. Only perform step 3 after steps 1 and 2 have both been completed.

Procedure

1. Define a local queue with the same name as the Version 6.0 stream.
2. Define a local topic with the same name as the Version 6.0 stream.
3. Add the name of the queue to the namelist, SYSTEM.QPUBSUB.QUEUE.NAMELIST
4. Repeat for all queue managers at Version 7.1 or above that are in the publish/subscribe hierarchy.

Adding 'Sport'

In the example of sharing the stream 'Sport', Version 6.0 and Version 7.1 queue managers are working in the same publish/subscribe hierarchy. The Version 6.0 queue managers share a stream called 'Sport'. The example shows how to create a queue and a topic on Version 7.1 queue managers called 'Sport', with a topic string 'Sport' that is shared with the version 6 stream 'Sport'.

A Version 7.1 publish application, publishing to topic 'Sport', with topic string 'Soccer/Results', creates the resultant topic string 'Sport/Soccer/Results' . On Version 7.1 queue managers, subscribers to topic 'Sport', with topic string 'Soccer/Results' receive the publication.

On Version 6.0 queue managers, subscribers to stream 'Sport', with topic string 'Soccer/Results' receive the publication.

```
runmqsc QM1
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.
define qlocal('Sport')
     1 : define qlocal('Sport')
```

```
AMQ8006: WebSphere MQ queue created.
define topic('Sport') topicstr('Sport')
     2 : define topic('Sport') topicstr('Sport')
AMQ8690: WebSphere MQ topic created.
alter namelist(SYSTEM.QPUBSUB.QUEUE.NAMELIST) NAMES('Sport', 'SYSTEM.BROKER.DEFAULT.STREAM', 'SYSTEM.BROKER.ADMIN.STREAM
     3 : alter namelist(SYSTEM.QPUBSUB.QUEUE.NAMELIST) NAMES('Sport', 'SYSTEM.BROKER.DEFAULT.STREAM', 'SYSTEM.BROKER.ADM
AMQ8551: WebSphere MQ namelist changed.
```

Note: You need both to provide the existing names in the namelist object, as well as the new names that you are adding, to the **alter namelist** command.

What to do next

Information about the stream is passed to other brokers in the hierarchy.

If a broker is Version 6.0, administer it as a Version 6.0 broker. That is, you have a choice of creating the stream queue manually, or letting the broker create the stream queue dynamically when it is needed. The queue is based on the model queue definition, SYSTEM.BROKER.MODEL.STREAM.

If a broker is Version 7.1, you must configure each Version 7.1 queue manager in the hierarchy manually.

Deleting a stream

You can delete a stream from a IBM WebSphere MQ Version 7.1, or later, queue manager.

Before you begin

The use of queued publish/subscribe is deprecated in IBM WebSphere MQ Version 7.1.

Before deleting a stream you must ensure that there are no remaining subscriptions to the stream and quiesce all applications that use the stream. If publications continue to flow to a deleted stream, it takes a lot of administrative effort to restore the system to a cleanly working state.

About this task

For instructions on deleting the stream from any Version 6.0 queue managers it is connected to, see Deleting a stream.

Procedure

1. Find all the connected brokers that host this stream.
2. Cancel all subscriptions to the stream on all the brokers.
3. Remove the queue (with the same name as the stream) from the namelist, SYSTEM.QPUBSUB.QUEUE.NAMELIST.
4. Delete or purge all the messages from the queue with the same name as the stream.
5. Delete the queue with the same name as the stream.
6. Delete the associated topic object.

What to do next

1. Repeat steps 3 to 5 on all the other connected Version 7.1, or later, queue managers hosting the stream.
2. Remove the stream from all other connected Version 6.0, or earlier, queue managers.

Adding a subscription point

How to add a subscription point that was not migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker by **migmbbrk**. Extend an existing queued publish/subscribe application that you have migrated from IBM WebSphere MQ Event Broker or IBM WebSphere MQ Message Broker with a new subscription point.

Before you begin

1. Complete the migration from IBM WebSphere MQ Event Broker and IBM WebSphere MQ Message Broker Version 6.0 to IBM WebSphere MQ Version 7.1.
2. Check that the subscription point is not already defined in SYSTEM.QPUBSUB.SUBPOINT.NAMELIST.
3. Check if there is a topic object or a topic string with the same name as the subscription point.

About this task

Existing IBM WebSphere MQ Event Broker applications use subscription points. New IBM WebSphere MQ Version 7.1 applications do not use subscription points, but they can interoperate with existing applications that do, using the subscription point migration mechanism.

A subscription point might not have been migrated by **migmbbrk**, if the subscription point was not in use at the time of the migration.

You might want to add a subscription point to existing queued publish/subscribe programs migrated from IBM WebSphere MQ Event Broker.

Subscription points do not work with queued publish/subscribe programs that use MQRFH1 headers, which have been migrated from IBM WebSphere MQ Version 6.0, or earlier.

There is no need to add subscription points to use integrated publish/subscribe applications written for IBM WebSphere MQ Version 7.1.

Procedure

1. Add the name of the subscription point to SYSTEM.QPUBSUB.SUBPOINT.NAMELIST.
   - On z/OS, the **NLTYPE** is NONE, the default.
   - Repeat the step on every queue manager that is connected in the same publish/subscribe topology.
2. Add a topic object, preferably giving it the name of the subscription point, with a topic string matching the name of the subscription point.
   - If the subscription point is in a cluster, add the topic object as a cluster topic on the cluster topic host.
   - If a topic object exists with the same topic string as the name of the subscription point, use the existing topic object. You must understand the consequences of the subscription point reusing an existing topic. If the existing topic is part of an existing application, you must resolve the collision between two identically named topics.
   - If a topic object exists with the same name as the subscription point, but a different topic string, create a topic with a different name.
3. Set the **Topic** attribute WILDCARD to the value BLOCK.

   Blocking subscriptions to # or * isolates wildcard subscriptions to subscription points, see Wildcards and subscription points.
4. Set any attributes that you require in the topic object.

Example

The example shows a **runmqsc** command file that adds two subscription points, USD and GBP.

```
DEFINE TOPIC(USD) TOPICSTR(USD)
DEFINE TOPIC(GBP) TOPICSTR(GBP) WILDCARD(BLOCK)
ALTER NL(SYSTEM.QPUBSUB.SUBPOINT.NAMELIST) NAMES(SYSTEM.BROKER.DEFAULT.SUBPOINT, USD, GBP)
```

Note:

1. Include the default subscription point in the list of subscription points added using the **ALTER** command. **ALTER** deletes existing names in the namelist.
2. Define the topics before altering the namelist. The queue manager only checks the namelist when the queue manager starts and when the namelist is altered.

Connect a queue manager to a broker hierarchy

You can connect a local queue manager to a parent queue manager to modify a broker hierarchy.

Before you begin

1. Enable queued publish/subscribe mode. See Starting queued publish/subscribe.
2. This change is propagated to the parent queue manager using a connection. There are two ways to establish the connection.
   - Connect the queue managers to a cluster, see Adding a queue manager to a cluster
   - Establish a point-to-point channel connection using a transmission queue, or queue manager alias, with the same name as the parent queue manager. For more information about how to establish a point-to-point channel connection, see WebSphere MQ distributed-messaging techniques.

About this task

Use the `ALTER QMGR PARENT` *`(PARENT_NAME)`* runmqsc command to connect children to parents.

Distributed publish/subscribe is implemented by using queue manager clusters and clustered topic definitions. For interoperability with and WebSphere Message Broker and WebSphere Event Broker and earlier, you can also connect or later queue managers to a broker hierarchy as long as queued publish/subscribe mode is enabled.

Procedure

ALTER QMGR PARENT(PARENT)

Example

The first example shows how to attach QM2 as a child of QM1, and then querying QM2 for its connection:

```
C:>runmqsc QM2
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM2
alter qmgr parent(QM1)
     1 : alter qmgr parent(QM1)
AMQ8005: WebSphere MQ queue manager changed.
display pubsub all
     2 : display pubsub all
AMQ8723: Display pub/sub status details.
   QMNAME(QM2)                            TYPE(LOCAL)
   STATUS(ACTIVE)
AMQ8723: Display pub/sub status details.
   QMNAME(QM1)                            TYPE(PARENT)
   STATUS(ACTIVE)
```

The next example shows the result of querying QM1 for its connections:

```
C:\Documents and Settings\Admin>runmqsc QM1
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.
display pubsub all
     2 : display pubsub all
AMQ8723: Display pub/sub status details.
   QMNAME(QM1)                             TYPE(LOCAL)
   STATUS(ACTIVE)
AMQ8723: Display pub/sub status details.
   QMNAME(QM2)                             TYPE(CHILD)
   STATUS(ACTIVE)
```

What to do next

You can define topics on one broker or queue manager that are available to publishers and subscribers on the connected queue managers. For more information, see Defining an administrative topic

Related information:

../../../com.ibm.mq.pla.doc/q005080_.dita

DISPLAY PUBSUB

../../../com.ibm.mq.pla.doc/q004870_.dita

Disconnect a queue manager from a broker hierarchy

Disconnect a child queue manager from a parent queue manager in a broker hierarchy.

About this task

Use the **ALTER QMGR** command to disconnect a queue manager from a broker hierarchy. You can disconnect a queue manager in any order at any time.

The corresponding request to update the parent is sent when the connection between the queue managers is running.

Procedure

```
ALTER QMGR PARENT('')
```

Example

```
C:\Documents and Settings\Admin>runmqsc QM2
5724-H72 (C) Copyright IBM Corp. 1994, 2008.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM2.
    1 : alter qmgr parent('')
AMQ8005: WebSphere MQ queue manager changed.
    2 : display pubsub type(child)
AMQ8147: WebSphere MQ object  not found.
display pubsub type(parent)
    3 : display pubsub type(parent)
AMQ8147: WebSphere MQ object  not found.
```

What to do next

You can delete any streams, queues and manually defined channels that are no longer needed.

Configuring a queue manager cluster

Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

Before you begin

For an introduction to clustering concepts, see the following topics:

- How clusters work
- "Comparison of clustering and distributed queuing" on page 888
- "Components of a cluster" on page 890

When you are designing your queue manager cluster you have to make some decisions. You must first decide which queue managers in the cluster are to hold the full repositories of cluster information. Any queue manager you create can work in a cluster. You can choose any number of queue managers for this purpose but the ideal number is two. For information about selecting queue managers to hold the full repositories, see "How to choose cluster queue managers to hold full repositories" on page 904.

See the following topics for more information about designing your cluster:

- "Organizing a cluster" on page 906
- "Cluster naming conventions" on page 906
- "Queue-sharing groups and clusters" on page 907
- "Overlapping clusters" on page 908

Example

The smallest possible cluster contains only two queue managers. In this case both queue managers contain full repositories. You need only a few definitions to set up the cluster, and yet there is a high degree of autonomy at each queue manager.

Figure 128 on page 886 shows a cluster called `DEMOCLSTR` with two queue managers called `QM1` and `QM2`.

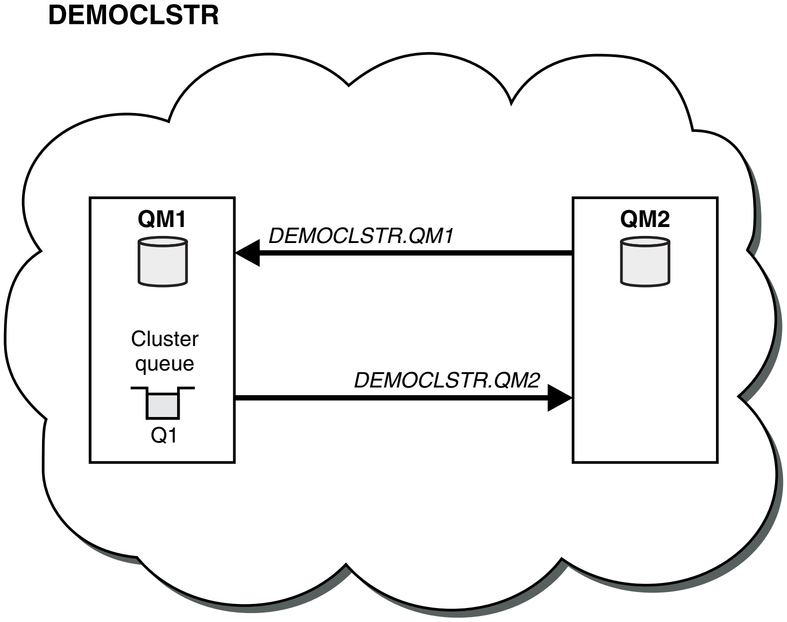


Figure 128. A small cluster of two queue managers

- The queue managers have long names such as LONDON and NEWYORK. The same names are used in the advanced and workload balancing tasks. On WebSphere MQ for z/OS, queue-manager names are limited to four characters.
- The names of the queue managers imply that each queue manager is on a separate machine. You could perform these tasks with all the queue managers on the same machine.
- The tasks use WebSphere MQ script commands as they would be entered by the system administrator using **MQSC** commands. There are other ways of entering commands, including using the easier WebSphere MQ Explorer. The point of using WebSphere MQ script commands is to demonstrate what WebSphere MQ commands are used in the tasks.

For instructions on setting up a similar example cluster, see “Setting up a new cluster” on page 913.

What to do next

See the following topics for more information about configuring and working with clusters:
- “Establishing communication in a cluster” on page 910
- “Managing IBM WebSphere MQ clusters” on page 912
- “Routing messages to and from clusters” on page 975
- “Using clusters for workload management” on page 988

For more information to help you configure your cluster, see “Clustering tips” on page 909.

Related information:

Clusters

Access control and multiple cluster transmission queues

Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, or checking against local profiles for the cluster queue, or cluster queue manager.

IBM WebSphere MQ gives you the choice of checking locally, or locally and remotely, that a user has permission to put a message to a remote queue. A typical IBM WebSphere MQ application uses local checking only, and relies on the remote queue manager trusting the access checks made on the local queue manager. If remote checking is not used, the message is put to the target queue with the authority of the remote message channel process. To use remote checking you must set the put authority of the receiving channel to context security.

The local checks are made against the queue that the application opens. In distributed queueing, the application usually opens a remote queue definition, and access checks are made against the remote queue definition. If the message is put with a full routing header, the checks are made against the transmission queue. If an application opens a cluster queue that is not on the local queue manager, there is no local object to check. The access control checks are made against the cluster transmission queue, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. Even with multiple cluster transmission queues, from Version 7.5, local access control checks for remote cluster queues are made against `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

The choice of local or remote checking is a choice between two extremes. Checking remotely is fine-grained. Every user must have an access control profile on every queue manager in the cluster to put to any cluster queue. Checking locally is coarse-grained. Every user needs only one access control profile for the cluster transmission queue on the queue manager they are connected to. With that profile, they can put a message to any cluster queue on any queue manager in any cluster.

Since Version 7.1, administrators have another way to set up access control for cluster queues. You can create a security profile for a cluster queue on any queue manager in the cluster using the **`setmqaut`** command. The profile takes affect if you open a remote cluster queue locally, specifying only the queue name. You can also set up a profile for a remote queue manager. If you do so, the queue manager can check the profile of a user that opens a cluster queue by providing a fully qualified name.

The new profiles work only if you change the queue manager stanza, **`ClusterQueueAccessControl`** to `RQMName`. The default is `Xmitq`. You must create profiles for all the cluster queues existing applications use cluster queues. If you change the stanza to `RQMName` without creating profiles the applications are likely to fail.

Tip: The changes made to cluster queue accessing checking in Version 7.1 do not apply to remote queueing. Access checks are still made against local definitions. The changes mean that you can follow the same approach to configure access checking on cluster queues and cluster topics.

Related concepts:

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

Comparison of clustering and distributed queuing

Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

If you do not use clusters, your queue managers are independent and communicate using distributed queuing. If one queue manager needs to send messages to another, you must define:

- A transmission queue
- A channel to the remote queue manager

Figure 129 shows the components required for distributed queuing.

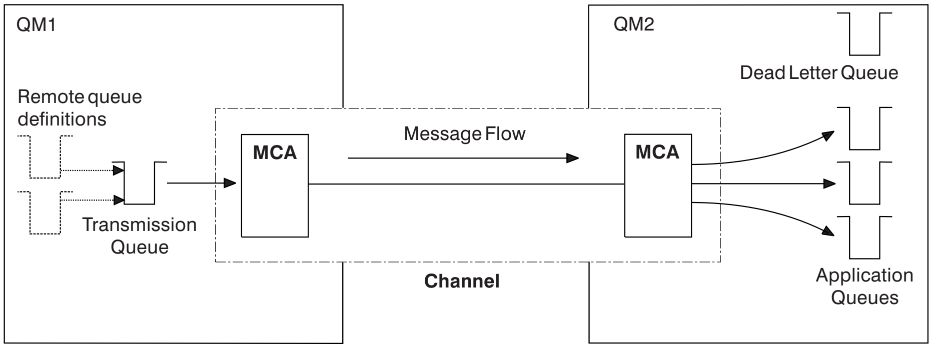


Figure 129. Distributed queuing

If you group queue managers in a cluster, queues on any queue manager are available to any other queue manager in the cluster. Any queue manager can send a message to any other queue manager in the same cluster without explicit definitions. You do not provide channel definitions, remote-queue definitions, or transmission queues for each destination. Every queue manager in a cluster has a single transmission queue from which it can transmit messages to any other queue manager in the cluster. Each queue manager in a cluster needs to define only:

- One cluster-receiver channel on which to receive messages
- One cluster-sender channel with which it introduces itself and learns about the cluster

Definitions to set up a cluster versus distributed queuing

Look at Figure 130 on page 889, which shows four queue managers each with two queues. Consider how many definitions are needed to connect these queue managers using distributed queuing. Compare how many definitions are needed to set up the same network as a cluster.

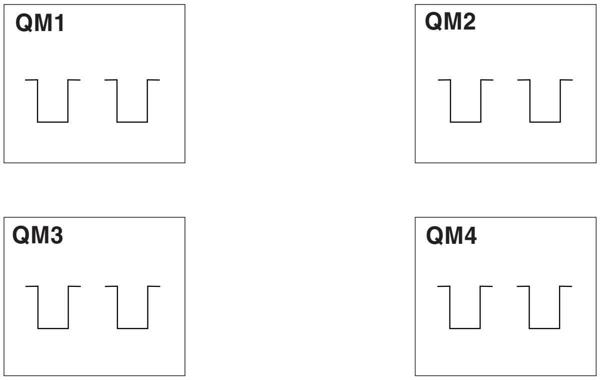


Figure 130. A network of four queue managers

Definitions to set up a network using distributed queuing

To set up the network shown in Figure 129 on page 888 using distributed queuing, you might have the following definitions:

Table 125. Definitions for distributed queuing

Description	Number per queue manager	Total number
A sender-channel definition for a channel on which to send messages to every other queue manager	3	12
A receiver-channel definition for a channel on which to receive messages from every other queue manager	3	12
A transmission-queue definition for a transmission queue to every other queue manager	3	12
A local-queue definition for each local queue	2	8
A remote-queue definition for each remote queue to which this queue manager wants to put messages	6	24

You might reduce this number of definitions by using generic receiver-channel definitions. The maximum number of definitions could be as many as 17 on each queue manager, which is a total of 68 for this network.

Definitions to set up a network using clusters

To set up the network shown in Figure 129 on page 888 using clusters you need the following definitions:

Table 126. Definitions for clustering

Description	Number per queue manager	Total number
A cluster-sender channel definition for a channel on which to send messages to a repository queue manager	1	4
A cluster-receiver channel definition for a channel on which to receive messages from other queue managers in the cluster	1	4
A local-queue definition for each local queue	2	8

To set up this cluster of queue managers (with two full repositories), you need four definitions on each queue manager, a total of sixteen definitions altogether. You also need to alter the queue-manager definitions for two of the queue managers, to make them full repository queue managers for the cluster.

Only one CLUSSDR and one CLUSRCVR channel definition is required. When the cluster is defined, you can add or remove queue managers (other than the repository queue managers) without any disruption to the other queue managers.

Using a cluster reduces the number of definitions required to set up a network containing many queue managers.

With fewer definitions to make there is less risk of error:

- Object names always match, for example the channel name in a sender-receiver pair.
- The transmission queue name specified in a channel definition always matches the correct transmission queue definition or the transmission queue name specified in a remote queue definition.
- A QREMOTE definition always points to the correct queue at the remote queue manager.

Once a cluster is set up, you can move cluster queues from one queue manager to another within the cluster without having to do any system management work on any other queue manager. There is no chance of forgetting to delete or modify channel, remote-queue, or transmission-queue definitions. You can add new queue managers to a cluster without any disruption to the existing network.

Components of a cluster

Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

See the following subtopics for information about each of the cluster components:

Related concepts:

"Comparison of clustering and distributed queuing" on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

"Managing IBM WebSphere MQ clusters" on page 912
You can create, extend, and maintain IBM WebSphere MQ clusters.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

"Setting up a new cluster" on page 913
Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Related information:

Clusters

Cluster repository

A repository is a collection of information about the queue managers that are members of a cluster.

The repository information includes queue manager names, their locations, their channels, which queues they host, and other information. The information is stored in the form of messages on a queue called SYSTEM.CLUSTER.REPOSITORY.QUEUE. The queue is one of the default objects. It is defined when you create a WebSphere MQ queue manager, except on WebSphere MQ for z/OS.

Typically, two queue managers in a cluster hold a full repository. The remaining queue managers all hold a partial repository.

Full repository and partial repository

A queue manager that hosts a complete set of information about every queue manager in the cluster has a full repository. Other queue managers in the cluster have partial repositories containing a subset of the information in the full repositories.

A partial repository contains information about only those queue managers with which the queue manager needs to exchange messages. The queue managers request updates to the information they need, so that if it changes, the full repository queue manager sends them the new information. For much of the time, a partial repository contains all the information a queue manager needs to perform within the cluster. When a queue manager needs some additional information, it makes inquiries of the full repository and updates its partial repository. The queue managers use a queue called `SYSTEM.CLUSTER.COMMAND.QUEUE` to request and receive updates to the repositories. This queue is one of the default objects.

Cluster queue manager

A cluster queue manager is a queue manager that is a member of a cluster.

A queue manager can be a member of more than one cluster. Each cluster queue manager must have a name that is unique throughout all the clusters of which it is a member.

A cluster queue manager can host queues, which it advertises to the other queue managers in the cluster. A cluster queue manager does not have to host or advertise any queues. It can feed messages into the cluster and receive only responses that are directed explicitly to it, and not to advertised queues.

In WebSphere MQ for z/OS, a cluster queue manager can be a member of a queue-sharing group. In this case, it shares its queue definitions with other queue managers in the same queue-sharing group.

Cluster queue managers are autonomous. They have full control over queues and channels that they define. Their definitions cannot be modified by other queue managers (other than queue managers in the same queue-sharing group). Repository queue managers do not control the definitions in other queue managers in the cluster. They hold a complete set of all the definitions, for use when required. A cluster is a federation of queue managers.

After you create or alter a definition on a cluster queue manager, the information is sent to the full repository queue manager. Other repositories in the cluster are updated later.

Full Repository queue manager

A full repository queue manager is a cluster queue manager that holds a full representation of the cluster's resources. To ensure availability, set up two or more full repository queue managers in each cluster. Full repository queue managers receive information sent by the other queue managers in the cluster and update their repositories. They send messages to each other to be sure that they are both kept up to date with new information about the cluster.

Queue managers and repositories

Every cluster has at least one (preferably two) queue managers holding full repositories of information about the queue managers, queues, and channels in a cluster. These repositories also contain requests from the other queue managers in the cluster for updates to the information.

The other queue managers each hold a partial repository, containing information about the subset of queues and queue managers with which they need to communicate. The queue managers build up their partial repositories by making inquiries when they first need to access another queue or queue manager. They request that they are notified of any new information concerning that queue or queue manager.

Each queue manager stores its repository information in messages on a queue called SYSTEM.CLUSTER.REPOSITORY.QUEUE. The queue managers exchange repository information in messages on a queue called SYSTEM.CLUSTER.COMMAND.QUEUE.

Each queue manager that joins a cluster defines a cluster-sender, CLUSSDR, channel to one of the repositories. It immediately learns which other queue managers in the cluster hold full repositories. From then on, the queue manager can request information from any of the repositories. When the queue manager sends information to the chosen repository, it also sends information to one other repository (if there is one).

A full repository is updated when the queue manager hosting it receives new information from one of the queue managers that is linked to it. The new information is also sent to another repository, to reduce the risk of it being delayed if a repository queue manager is out of service. Because all the information is sent twice, the repositories have to discard duplicates. Each item of information carries a sequence number, which the repositories use to identify duplicates. All repositories are kept in step with each other by exchanging messages.

Cluster queues

A cluster queue is a queue that is hosted by a cluster queue manager and made available to other queue managers in the cluster.

Define a cluster queue as a local queue on the cluster queue manager where the queue is hosted. Specify the name of the cluster the queue belongs to. The following example shows a **runmqsc** command to define a cluster queue with the CLUSTER option:

```
DEFINE QLOCAL(Q1) CLUSTER(SALES)
```

A cluster queue definition is advertised to other queue managers in the cluster. The other queue managers in the cluster can put messages to a cluster queue without needing a corresponding remote-queue definition. A cluster queue can be advertised in more than one cluster by using a cluster namelist.

When a queue is advertised, any queue manager in the cluster can put messages to it. To put a message, the queue manager must find out, from the full repositories, where the queue is hosted. Then it adds some routing information to the message and puts the message on a cluster transmission queue.

A cluster queue can be a queue that is shared by members of a queue-sharing group in IBM WebSphere MQ for z/OS.

Binding

You can create a cluster in which more than one queue manager hosts an instance of the same cluster queue. Make sure that all the messages in a sequence are sent to the same instance of the queue. You can bind a series of messages to a particular queue by using the MQOO_BIND_ON_OPEN option on the MQOPEN call.

Cluster transmission queues

Except on z/OS, a queue manager can store messages for other queue managers in a cluster on multiple transmission queues. You can configure a queue manager to store messages on multiple cluster transmission queues in two different ways. If you set the queue manager attribute DEFCLXQ to CHANNEL, a different cluster transmission queue is created automatically from SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE for each cluster-sender channel. If you set the CLCHNAME transmission queue option to match one or more cluster-senders channel, the queue manager can store messages for the matching channels on that transmission queue.

A message for a cluster queue on a different queue manager is placed upon a cluster transmission queue before being sent. A cluster-sender channel transfers the messages from a cluster transmission queue to cluster-receiver channels on other queue managers. By default, one system defined cluster transmission queue holds all the messages that are to be transferred to other cluster queue managers. The queue is called `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. A queue manager that is part of a cluster can send messages on this cluster transmission queue to any other queue manager in the same cluster.

A definition for the single `SYSTEM.CLUSTER.TRANSMIT.QUEUE` queue is created by default on every queue manager except on z/OS.

On platforms other than z/OS, you can configure a queue manager to transfer messages to other clustered queue managers using multiple transmission queues. You can define additional cluster transmission queues manually, or have the queue manager create the queues automatically.

To have the queues created automatically by the queue manager, change the queue manager attribute `DEFCLXQ` from `SCTQ` to `CHANNEL`. The result is the queue manager creates an individual cluster transmission queue for each cluster-sender channel that is created. The transmission queues are created as permanent dynamic queues from the model queue, `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`. The name of each permanent dynamic queue is `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*. The name of the cluster-sender channel that each permanent dynamic cluster transmit queue is associated with is set in the local transmission queue attribute `CLCHNAME`. Messages for remote clustered queue managers are placed on the permanent dynamic cluster transmission queue for the associated cluster-sender channel, rather than on `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

To create the cluster transmission queues manually, create a local queue with the `USAGE` attribute set to `XMITQ`, and the `CLCHNAME` attribute set to a generic channel name that resolves to one or more cluster-sender channels; see ClusterChannelName. If you create cluster transmission queues manually, you have the choice of associating the transmission queue with a single cluster-sender channel, or with multiple cluster-sender channels. The `CLCHNAME` attribute is a generic-name, which means you can place multiple wildcard characters, "*", in the name.

Except for the initial cluster-sender channels that you create manually to connect a queue manager to a full repository, cluster-sender channels are created automatically. They are created automatically when there is a message to transfer to a cluster queue manager. They are created with the same name as the name of the cluster-receiver channel that receives cluster messages for that particular cluster on the destination queue manager.

If you follow a naming convention for cluster-receiver channels, it is possible to define a generic value for `CLCHNAME` that filters different kinds of cluster messages to different transmission queues. For example, if you follow the naming convention for cluster-receiver channels of *`ClusterName.QmgrName`*, then the generic name *`ClusterName`*`.*` filters messages for different clusters onto different transmission queues. You must define the transmission queues manually, and set `CLCHNAME` in each transmission queue to *`ClusterName`*`.*`.

Changes to the association of cluster transmission queues to cluster-sender channels do not take immediate effect. The currently associated transmission queue that a cluster-sender channel is servicing might contain messages that are in the process of being transferred by the cluster-sender channel. Only when no messages on the currently associated transmission queue are being processed by a cluster-sender channel, can the queue manager change the association of the cluster-sender channel a different transmission queue. This can occur either when no messages remain on the transmission queue to be processed by the cluster-sender channel, or when processing of messages is suspended and the cluster-sender channel has no "in-flight" messages. When this happens, any unprocessed messages for the cluster-sender channel are transferred to the newly associated transmission queue, and the association of the cluster-sender channel changes.

You can create a remote queue definition that resolves to a cluster transmission queue. In the definition, queue manager QMX is in the same cluster as the local queue manager, and there is no transmission queue, QMX.

```
DEFINE QREMOTE(A) RNAME(B) RQMNAME(QMX)
```

During queue name resolution, the cluster transmission queue takes precedence over the default transmission queue. A message put to A is stored on the cluster transmission queue and then sent to the remote queue B on QMX.

Queue managers can also communicate with other queue managers that are not part of a cluster. You must define channels and a transmission queue to the other queue manager, in the same way as in a distributed-queuing environment.

Note: Applications must write to queues that resolve to the cluster transmission queue, and must not write directly to the cluster transmission queue.

Auto definition of remote queues

A queue manager in a cluster does not need a remote-queue definition for remote queues in the cluster. The cluster queue manager finds the location of a remote queue from the full repository. It adds routing information to the message and puts it on the cluster transmission queue. WebSphere MQ automatically creates a definition equivalent to a remote-queue definition so that the message can be sent.

You cannot alter or delete an automatically created remote-queue definition. However, by using the DISPLAY QUEUE **runmqsc** command with the CLUSINFO attribute, you can view all of the local queues on a queue manager as well as all of the cluster queues, including cluster queues on remote queue managers. For example:

```
DISPLAY QUEUE(*) CLUSINFO
```

Related information:

ClusterChannelName (MQCHAR20)

Cluster channels

You must define cluster-receiver and cluster-sender channels for queue managers in your cluster. Special considerations apply to full repositories.

Within clusters, messages are distributed between cluster queue managers on a special type of channel for which you need cluster-receiver channel definitions and cluster-sender channel definitions.

Cluster-sender channel: CLUSSDR

Manually define a cluster-sender channel to a full repository at each queue manager in the cluster. The cluster-sender definition enables the queue manager to make its initial contact with the cluster. It names the full repository queue manager to which the queue manager preferentially chooses to send cluster information. The cluster-sender channel is used to notify the repository of any changes to the status of the queue manager. For example, if a queue is added or removed. It is also used to transmit messages.

The full repository queue managers themselves have cluster-sender channels that point to each other. They use them to communicate cluster status changes to each other.

It is of little importance which full repository a CLUSSDR channel definition points to. Once the initial contact has been made, further cluster queue manager objects are defined automatically as necessary. The queue manager can send cluster information to every full repository, and messages to every queue manager.

The CLUSSDR definitions made on the full repository queue managers are special. All the updates exchanged by the full repositories flow exclusively on these channels. The administrator controls the network of full repositories explicitly. The administrator must define a CLUSSDR channel from every full repository queue manager to every other full repository queue manager in the cluster. The administrator must make the CLUSSDR definitions on full repository queue managers manually and not leave them to be auto-defined.

Cluster sender channels must only be defined to connect a partial repository to a full repository, or to connect two full repositories together. Manually configuring a CLUSSDR channel that addresses a partial repository, or a queue manager not in the cluster, leads to error messages, such as AMQ9427 and AMQ9428 being issued.

Although this might sometimes be unavoidable as a temporary situation (for example when modifying the location of a full repository) the incorrect definitions should be deleted as soon as possible to stop these errors being issued.

Cluster-receiver channel: CLUSRCVR

A cluster-receiver channel definition defines the end of a channel on which a cluster queue manager can receive messages from other queue managers in the cluster.

A cluster-receiver channel can also carry information about the cluster-information destined for the local repository. By defining the cluster-receiver channel, the queue manager shows the other cluster queue managers that it is available to receive messages. You need at least one cluster-receiver channel for each cluster queue manager.

A CLUSRCVR definition enables other queue managers to auto-define corresponding CLUSSDR channel definitions.

Related concepts:

"Auto-definition of cluster channels"
A queue manager must have a definition for a cluster-sender channel before it can send a message to a remote destination. After you introduce a queue manager to a cluster by making its initial CLUSSDR and CLUSRCVR definitions, WebSphere MQ automatically makes cluster-sender channel definitions when they are needed. You cannot modify auto-defined cluster-sender channels. You can modify their behavior using a channel auto-definition exit.

Auto-definition of cluster channels:

A queue manager must have a definition for a cluster-sender channel before it can send a message to a remote destination. After you introduce a queue manager to a cluster by making its initial CLUSSDR and CLUSRCVR definitions, WebSphere MQ automatically makes cluster-sender channel definitions when they are needed. You cannot modify auto-defined cluster-sender channels. You can modify their behavior using a channel auto-definition exit.

When both the cluster-sender end and the cluster-receiver end of a channel are defined, the channel is started. An auto-defined channel remains active until it is no longer needed and is shut down using the normal disconnect-interval rules.

Auto-defined cluster-sender channels take their attributes from the corresponding cluster-receiver channel definition on the receiving queue manager. Even if there is a manually defined cluster-sender channel, its attributes are automatically modified to ensure that they match the corresponding cluster-receiver definition. Suppose, for example that you define a CLUSRCVR without specifying a port number in the CONNAME parameter, and manually define a CLUSSDR that does specify a port number. When the auto-defined CLUSSDR replaces the manually defined one, the port number (taken from the CLUSRCVR) becomes blank. The default port number is used and the channel fails.

You cannot modify an auto-defined cluster-sender definition.

You cannot see automatically defined channels using the DISPLAY CHANNEL **runmqsc** command. To see the auto-defined channels use the command:

```
DISPLAY CLUSQMGR(qMgrName)
```

To display the status of the auto-defined CLUSSDR channel corresponding to the CLUSRCVR channel definition you created, use the command:

```
DISPLAY CHSTATUS(channelname)
```

You can use the WebSphere MQ channel auto-definition exit if you want to write a user exit program to customize a cluster-sender channel or cluster-receiver channel. You might use the channel auto-definition exit in a cluster environment to:

- Tailor communications definitions, that is, SNA LU6.2 names
- Add or remove other exits, for example, security exits
- Change the names of channel exits. You must change the name of a CLUSSDR channel exit because the CLUSSDR channel exit name is auto-generated from the CLUSRCVR channel definition. The auto-generated name might be wrong, and almost certainly is wrong if the two ends of the channel are on different platforms. The format of exit names is different on different platforms. For example, on Windows it is, SCYEXIT('*drive*:*path**library*(secexit)').

  Exit names on platforms other than z/OS are of the general form *path*/*library*(*function*). If *function* is present, up to eight chars of that are used. Otherwise, *library*, truncated to eight chars is used. For example,

  - /var/mqm/exits/myExit.so(MsgExit) converts to MSGEXIT
  - /var/mqm/exits/myExit converts to MYEXIT
  - /var/mqm/exits/myExit.so(ExitLongName) converts to EXITLONG

To enable an outbound (TCP) channel to use a particular IP address, port or port range, use the channel attribute LOCLADDR. LOCLADDR is useful if you have more than one network card and you want a channel to use a specific one for outbound communications.

To specify a virtual IP address on CLUSSDR channels, use the IP address from the LOCLADDR on a manually defined CLUSSDR. To specify the port range, use the port range from the CLUSRCVR.

If a cluster needs to use LOCLADDR to get the outbound communication channels to bind to a specific IP address, you must write a Channel Auto-Definition Exit in order to force the LOCLADDR value into any of their automatically defined CLUSSDR channels, and you must specify it in the manually defined CLUSSDR channel.

Do not put an IP address in the LOCLADDR field of a CLUSRCVR channel, unless all queue managers are on the same server. The LOCLADDR IP address is propagated to the auto-defined CLUSSDR channels of all queue managers that connect using the CLUSRCVR channel.

Put a port number or port range in the LOCLADDR of a CLUSRCVR channel, if you want all the queue managers in a cluster to use a specific port or range of ports, for all their outbound communications

On distributed platforms, it is possible to set a default local address value that will be used for all sender channels that do not have a local address defined. The default value is defined by setting the MQ_LCLADDR environment variable prior to starting the queue manager. The format of the value matches that of MQSC attribute LOCLADDR.

Auto-defined cluster-sender channel definitions are not real channel objects. On platforms other than z/OS, the OAM (object authority manager) is not aware of their existence. If you try to issue start, stop, ping, reset, or resolve commands on auto-defined cluster-sender channels, the OAM checks to see whether you are authorized to perform the same action on the cluster-receiver channel for the cluster.

If the cluster needs to use PROPCTL to remove application headers such as RFH2 from messages going from a WebSphere MQ Version 7 queue manager to a queue manager on an earlier level of WebSphere MQ, you must write a Channel Auto-Definition Exit that forces PROPCTL to a value of NONE. The exit is necessary because cluster-sender channels have their definition based on the corresponding cluster-receiver channels. As the earlier level cluster-receiver channel does not have a PROPCTL attribute, the attribute is set to COMPAT by the auto cluster-sender channel. The attribute is set to COMPAT regardless of what is set on the manual cluster-sender channel.

Related information:

Local Address (LOCLADDR)

Default cluster objects:

Create the default cluster objects when using WebSphere MQ clusters. They are included in the set of default objects automatically created when you define a queue manager.

You can alter the default channel definitions in the same way as any other channel definition, by running MQSC or PCF commands.

Do not alter the default queue definitions, except for `SYSTEM.CLUSTER.HISTORY.QUEUE`.

`SYSTEM.CLUSTER.COMMAND.QUEUE`
Each queue manager in a cluster has a local queue called `SYSTEM.CLUSTER.COMMAND.QUEUE` which is used to transfer messages to the full repository. The message contains any new or changed information about the queue manager, or any requests for information about other queue managers. `SYSTEM.CLUSTER.COMMAND.QUEUE` is normally empty.

`SYSTEM.CLUSTER.HISTORY.QUEUE`
Each queue manager in a cluster has a local queue called `SYSTEM.CLUSTER.HISTORY.QUEUE`. `SYSTEM.CLUSTER.HISTORY.QUEUE` is used to store the history of cluster state information for service purposes.

In the default object settings, `SYSTEM.CLUSTER.HISTORY.QUEUE` is set to `PUT(ENABLED)`. To suppress history collection change the setting to `PUT(DISABLED)`.

`SYSTEM.CLUSTER.REPOSITORY.QUEUE`
Each queue manager in a cluster has a local queue called `SYSTEM.CLUSTER.REPOSITORY.QUEUE`. This queue is used to store all the full repository information. This queue is not normally empty.

`SYSTEM.CLUSTER.TRANSMIT.QUEUE`
Each queue manager has a definition for a local queue called `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. `SYSTEM.CLUSTER.TRANSMIT.QUEUE` is the default transmission queue for all messages to all queues and queue managers that are within clusters. You can change the default transmission queue for each cluster-sender channel to `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*, by changing the queue manager attribute `DEFXMITQ`. You cannot delete `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. It is also used to define authorization checks whether the default transmission queue that is used is `SYSTEM.CLUSTER.TRANSMIT.QUEUE` or `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*.

`SYSTEM.DEF.CLUSRCVR`
Each cluster has a default `CLUSRCVR` channel definition called `SYSTEM.DEF.CLUSRCVR`. `SYSTEM.DEF.CLUSRCVR` is used to supply default values for any attributes that you do not specify when you create a cluster-receiver channel on a queue manager in the cluster.

`SYSTEM.DEF.CLUSSDR`
Each cluster has a default `CLUSSDR` channel definition called `SYSTEM.DEF.CLUSSDR`. `SYSTEM.DEF.CLUSSDR` is used to supply default values for any attributes that you do not specify when you create a cluster-sender channel on a queue manager in the cluster.

Cluster transmission queues and cluster-sender channels

Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

When you display cluster-sender channel you see that it is associated with a transmission queue. At any point in time, a cluster-sender channel is associated with one transmission queue. If you change the configuration of the channel, it might switch to a different transmission queue next time it starts. Run the following MQSC command to display the transmission queues that cluster-sender channels are associated with:

```
DISPLAY CHSTATUS(*) WHERE(CHLTYPE EQ CLUSSDR)
```

```
AMQ8417: Display Channel Status details.
   CHANNEL(TO.QM2)                    CHLTYPE(CLUSSDR)
   CONNAME(9.146.163.190(1416))       CURRENT
   RQMNAME(QM2)                       STATUS(STOPPED)
   SUBSTATE( )                        XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
```

The transmission queue shown in the saved channel status of a stopped cluster-sender channel might change when the channel starts again. "Selection of default transmission queues by cluster-sender channels" describes the process of selecting a default transmission queue; "Selection of manually defined transmission queues by cluster-sender channels" on page 899 describes the process of selecting a manually defined transmission queue.

When any cluster-sender channel starts it rechecks its association with transmission queues. If the configuration of transmission queues, or the queue manager defaults, changes, it might reassociate the channel with a different transmission queue. If the channel restarts with a different transmission queue as a result of a configuration change, a process of transferring messages to the newly associated transmission queue takes place. "How the process to switch cluster-sender channel to a different transmission queue works" on page 900 describes the process of transferring a cluster-sender channel from one transmission queue to another.

The behavior of cluster-sender channels is different to sender and server channels. They remain associated with the same transmission queue until the channel attribute **XMITQ** is altered. If you alter the transmission queue attribute on a sender or server channel, and restart it, messages are not transferred from the old transmission queue to the new one.

Another difference between cluster-sender channels, and sender or server channels, is that multiple cluster-sender channels can open a cluster transmission queue, but only one sender or server channel can open a normal transmission queue. Until Version 7.5 cluster connections shared the single cluster transmission queue, SYSTEM.CLUSTER.TRANSMIT.QUEUE. From Version 7.5 onwards you have the option of cluster-sender channels not sharing transmission queues. Exclusivity is not enforced; it is an outcome of the configuration. You can configure the path a message takes in a cluster so that it does not share any transmission queues or channels with messages that flow between other applications. See "Clustering: Planning how to configure cluster transmission queues" on page 1012 and "Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934.

Selection of default transmission queues by cluster-sender channels

A cluster transmission queue is either a system default queue, with a name that starts with SYSTEM.CLUSTER.TRANSMIT, or a manually defined queue. A cluster-sender channel is associated with a cluster transmission queue in one of two ways: by the default cluster transmission queue mechanism, or by manual configuration.

The default cluster transmission queue is set as a queue manager attribute, **DEFCLXQ**. Its value is either SCTQ or CHANNEL. New and migrated queue managers are set to SCTQ. You can alter the value to CHANNEL.

If SCTQ is set, the default cluster transmission queue is SYSTEM.CLUSTER.TRANSMIT.QUEUE. Every cluster-sender channel can open this queue. The cluster-sender channels that do open the queue are the ones that are not associated with manually defined cluster transmission queues.

If CHANNEL is set, then the queue manager can create a separate permanent dynamic transmission queue for every cluster-sender channel. Each queue is named SYSTEM.CLUSTER.TRANSMIT.*ChannelName* and is created from the model queue, SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE. Each cluster-sender channel that is not associated with a manually defined cluster transmission queue is associated with a permanent-dynamic cluster transmission queue. The queue is created by the queue manager when it requires a separate cluster transmission queue for the cluster destination served by this cluster-sender channel, and no queue exists.

Some cluster destinations can be served by cluster-sender channels associated with manually defined transmission queues, and others by the default queue or queues. In the association of cluster-sender channels with transmission queues, the manually defined transmission queues always take precedence over the default transmission queues.

The precedence of cluster transmission queues is illustrated in Figure 131. The only cluster-sender channel not associated with a manually defined cluster transmission queue is CS.QM1. It is not associated with a manually defined transmission queue, because none of the channel names in the **CLCHNAME** attribute of the transmission queues match CS.QM1.

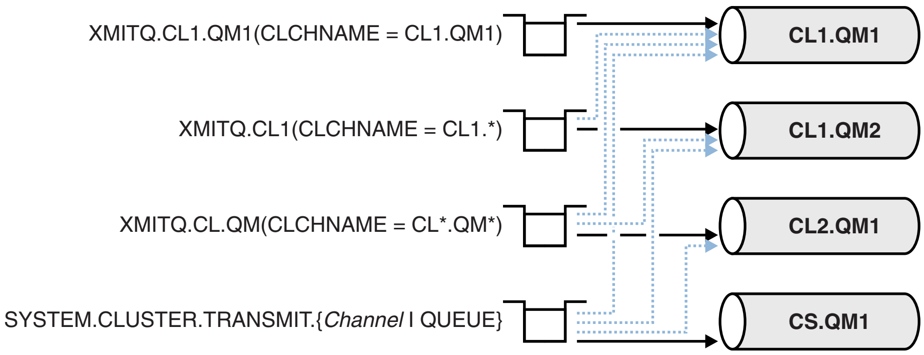

XMITQ.CL1.QM1(CLCHNAME = CL1.QM1)
XMITQ.CL1(CLCHNAME = CL1.*)
XMITQ.CL.QM(CLCHNAME = CL*.QM*)
SYSTEM.CLUSTER.TRANSMIT.{*Channel* I QUEUE}
CL1.QM1
CL1.QM2
CL2.QM1
CS.QM1

Figure 131. Transmission queue / cluster-sender channel precedence

Selection of manually defined transmission queues by cluster-sender channels

A manually defined queue has the transmission queue attribute **USAGE** attribute set to XMITQ, and the cluster channel name attribute **CLCHNAME** set to a specific or generic channel name.

If the name in the **CLCHNAME** queue attribute matches a cluster-sender channel name, the channel is associated with the queue. The name is either an exact match, if the name contains no wildcards, or it the best match, if the name contains wildcards.

If **CLCHNAME** definitions on multiple transmission queues match the same cluster-sender channel, the definitions are said to overlap. To resolve the ambiguity there is an order of precedence between matches. Exact matches always take precedence. Figure 131 shows associations between transmission queues and cluster-sender channels. The black arrows show actual associations, and the gray arrows, potential associations. The precedence order of transmission queues in Figure 131 is,

XMITQ.CL1.QM1
: The transmission queue XMITQ.CL1.QM1 has its **CLCHNAME** attribute set to CL1.QM1. The definition of the **CLCHNAME** attribute, CL1.QM1, has no wildcards, and takes precedence over any other CLCHNAME attributes, defined on other transmission queues, that match with wildcards. The queue manager stores any cluster message that is to be transferred by the CL1.QM1 cluster-sender channel on the

XMITQ.CL1.QM1 transmission queue. The only exception is if multiple transmission queues have their **CLCHNAME** attribute set to CL1.QM1. In that case, the queue manager stores messages for the CL1.QM1 cluster-sender channel on any of those queues. It selects a queue arbitrarily when the channel starts. It might select a different queue when the channel starts again.

XMITQ.CL1

The transmission queue XMITQ.CL1 has its **CLCHNAME** attribute set to CL1.*. The definition of the **CLCHNAME** attribute, CL1.*, has one trailing wildcard, which matches the name of any cluster-sender channel that starts with CL1.. The queue manager stores any cluster message that is to be transferred by any cluster-sender channel whose name begins with CL1. on the transmission queue XMITQ.CL1, unless there is a transmission queue with a more specific match, such as the queue XMITQ.CL1.QM1. One trailing wildcard makes the definition less specific than a definition with no wildcards, and more specific than a definition with multiple wildcards, or wildcards that are followed by more trailing characters.

XMITQ.CL.QM

XMITQ.CL.QM is the name of the transmission queue with its **CLCHNAME** attribute set to CL*.QM*. The definition of CL*.QM* has two wildcards, which match the name of any cluster-sender channel that starts with CL., and either includes or ends with QM. The match is less specific than a match with one wildcard.

SYSTEM.CLUSTER.TRANSMIT.*channelName*|QUEUE

If no transmission queue has a **CLCHNAME** attribute that matches the name of the cluster-sender channel that the queue manager is to use, then the queue manager uses the default cluster transmission queue. The default cluster transmission queue is either the single system cluster transmission queue, SYSTEM.CLUSTER.TRANSMIT.QUEUE, or a system cluster transmission queue that the queue manager created for a specific cluster-sender channel, SYSTEM.CLUSTER.TRANSMIT.*channelName*. Which queue is the default depends on the setting of the queue manager **DEFXMITQ** attribute.

Tip: Unless you have a clear need for overlapping definitions, avoid them as they can lead to complicated configurations that are hard to understand.

How the process to switch cluster-sender channel to a different transmission queue works

To change the association of cluster-sender channels with cluster transmission queues, change the **CLCHNAME** parameter of any transmission queue or the queue manager parameter **DEFCLXQ** at any time. Nothing happens immediately. Changes occur only when a channel starts. When it starts, it checks whether to continue forwarding messages from the same transmission queue. Three kinds of change alter the association of a cluster-sender channel with a transmission queue.

1. Redefining the **CLCHNAME** parameter of the transmission queue the cluster-sender channel is currently associated with to be less specific or blank, or deleting the cluster transmission queue when the channel is stopped.

   Some other cluster transmission queue might now be a better match for the channel name. Or, if no other transmission queues match the name of the cluster-sender channel, the association must revert to the default transmission queue.

2. Redefining the **CLCHNAME** parameter of any other cluster transmission queue, or adding a cluster transmission queue.

   The **CLCHNAME** parameter of another transmission queue might now be a better match for the cluster-sender channel than the transmission queue the cluster-sender channel is currently associated with. If the cluster-sender channel is currently associated with a default cluster transmission queue, it might become associated with a manually defined cluster transmission queue.

3. If the cluster-sender channel is currently associated with a default cluster transmission queue, changing the **DEFCLXQ** queue manager parameter.

If the association of a cluster-sender channel changes, when the channel starts it switches its association to the new transmission queue. During the switch, it ensures that no messages are lost. Messages are transferred to the new transmission queue in the order in which the channel would transfer the messages to the remote queue manager.

Remember: In common with any forwarding of messages in a cluster, you must put messages into groups to ensure that messages that must be delivered in order are delivered in order. On rare occasions, messages can get out of order in a cluster.

The switch process goes through the following transactional steps. If the switch process is interrupted, the current transactional step is resumed when the channel restarts again.

Step 1 - Process messages from the original transmission queue

The cluster-sender channel is associated with the new transmission queue, which it might share with other cluster-sender channels. Messages for the cluster-sender channel continue to be placed on the original transmission queue. A transitional switch process transfers messages from the original transmission queue onto the new transmission queue. The cluster-sender channel forwards the messages from the new transmission queue to the cluster-receiver channel. The channel status shows the cluster-sender channel still associated with the old transmission queue.

The switch process continues to transfer newly arrived messages as well. This step continues until the number of remaining messages to be forwarded by the switch process reaches zero. When the number of messages reaches zero, the procedure moves onto step 2.

During step 1, disk activity for the channel increases. Persistent messages are committed off the first transmission queue and onto the second transmission queue. This disk activity is in addition to messages being committed when they are placed on and removed from the transmission queue as part of transferring the messages normally. Ideally, no messages arrive during the switching process, so the transition can take place as quickly as possible. If messages do arrive, they are processed by the switch process.

Step 2 - Process messages from the new transmission queue

As soon as no messages remain on the original transmission queue for the cluster-sender channel, new messages are placed directly on the new transmission queue. The channel status shows the cluster-sender channel is associated with the new transmission queue. The following message is written to the queue manager error log: “AMQ7341 `The transmission queue for channel` *`ChannelName`* `is` *`QueueName`*.”

Multiple cluster transmission queues and cluster transmission queue attributes

You have a choice of forwarding cluster messages to different queue managers storing the messages on a single cluster transmission queue, or multiple queues. With one queue, you have one set of cluster transmission queue attributes to set and query, with multiple queues, you have multiple sets. For some attributes, having multiple sets is an advantage: for example querying queue depth tells you how many messages are waiting to be forwarded by one or a set of channels, rather than by all channels. For other attributes, having multiple sets is a disadvantage: for example, you probably do not want to configure the same access permissions for every cluster transmission queue. For this reason, access permissions are always checked against the profile for `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, and not against profiles for a particular cluster transmission queue. If you want to apply more granular security checks, see “Access control and multiple cluster transmission queues” on page 887.

Multiple cluster-sender channels and multiple transmission queues

A queue manager stores a message on a cluster transmission queue before forwarding it on a cluster-sender channel. It selects a cluster-sender channel that is connected to the destination for the message. It might have a choice of cluster-sender channels that all connect to the same destination. The destination might be the same physical queue, connected by multiple cluster-sender channels to a single queue manager. The destination might also be many physical queues with the same queue name, hosted on different queue managers in the same cluster. Where is a choice of cluster-sender channels connected to a destination, the workload balancing algorithm chooses one. The choice depends on a number of factors; see The cluster workload management algorithm.

In Figure 132 on page 903, CL1.QM1, CL1.QM2 and CS.QM1 are all channels that might lead to the same destination. For example, if you define Q1 in CL1 on QM1 and QM2 then CL1.QM1 and CL1.QM2 both provide routes to the same destination, Q1, on two different queue managers. If the channel CS.QM1 is also in CL1, it too is a channel that a message for Q1 can take. The cluster membership of CS.QM1 might be defined by a cluster namelist, which is why the channel name does not include a cluster name in its construction. Depending on the workload balancing parameters, and the sending application, some messages for Q1 might be placed on each of the transmission queues, XMITQ.CL1.QM1, XMITQ.CL1 and SYSTEM.CLUSTER.TRANSMIT.CS.QM1.

If you intend to separate message traffic, so that messages for the same destination do not share queues or channels with messages for different destinations, you must consider how to divide traffic onto different cluster-sender channels first, and then how to separate messages for a particular channel onto a different transmission queue. Cluster queues on the same cluster, on the same queue manager, normally share the same cluster channels. Defining multiple cluster transmission queues alone is not sufficient to separate cluster message traffic onto different queues. Unless you separate messages for different destination queues onto different channels, the messages share the same cluster transmission queue.

A straightforward way to separate the channels that messages take, is to create multiple clusters. On any queue manager in each cluster, define only one cluster queue. Then, if you define a different cluster-receiver channel for each cluster/queue manager combination, the messages for each cluster queue do not share a cluster channel with messages for other cluster queues. If you define separate transmission queues for the cluster channels, the sending queue manager stores messages for only one cluster queue on each transmission queue. For example, if you want two cluster queues not to share resources, you can either place them in different clusters on the same queue manager, or on different queue managers in the same cluster.

The choice of cluster transmission queue does not affect the workload balancing algorithm. The workload balancing algorithm chooses which cluster-sender channel to forward a message. It places the message on the transmission queue that is serviced by that channel. If the workload balancing algorithm is called on to choose again, for instance if the channel stops, it might be able to select a different channel to forward the message. If it does choose a different channel, and the new channel forwards messages from a different cluster transmission queue, the workload balancing algorithm transfers the message to the other transmission queue.

In Figure 132 on page 903, two cluster-sender channels, CS.QM1 and CS.QM2, are associated with the default system transmission queue. When the workload balancing algorithm stores a message on SYSTEM.CLUSTER.TRANSMIT.QUEUE, or any other cluster transmission queue, the name of the cluster-sender channel that is to forward the message is stored in the correlation ID of the message. Each channel forwards just those messages that match the correlation ID with the channel name.

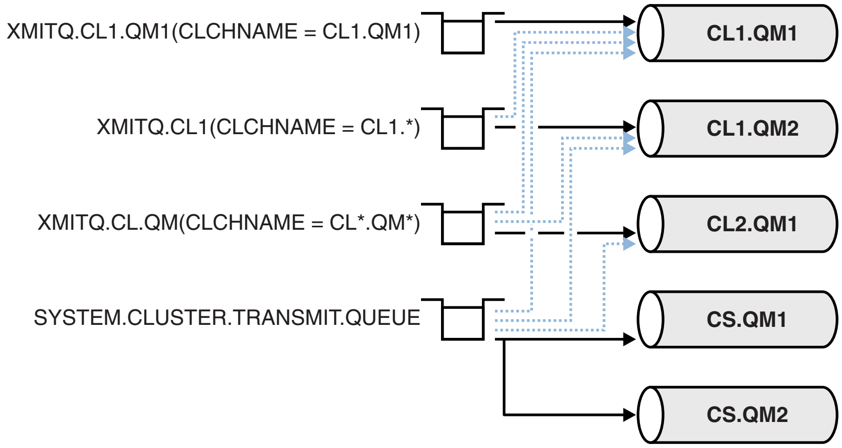

XMITQ.CL1.QM1(CLCHNAME = CL1.QM1)
XMITQ.CL1(CLCHNAME = CL1.*)
XMITQ.CL.QM(CLCHNAME = CL*.QM*)
SYSTEM.CLUSTER.TRANSMIT.QUEUE
CL1.QM1
CL1.QM2
CL2.QM1
CS.QM1
CS.QM2

Figure 132. Multiple cluster sender channels

If CS.QM1 stops, the messages on the transmission queue for that cluster-sender channel are examined. Those messages that can be forwarded by another channel are reprocessed by the workload balancing algorithm. Their correlation ID is reset to an alternative cluster-sender channel name. If the alternative cluster-sender channel is CS.QM2, the message remains on SYSTEM.CLUSTER.TRANSMIT.QUEUE. If the alternative channel is CL1.QM1, the workload balancing algorithm transfers the message to XMITQ.CL1.QM1. When the cluster-sender channel restarts, new messages, and messages that were not flagged for a different cluster-sender channel, are transferred by the channel again.

You might change the association between transmission queues and cluster-sender channels on a running system. You might change a CLCHNAME parameter on a transmission queue, or, change the **DEFCLXQ** queue manager parameter. When a channel that is affected by the change restarts, it starts the transmission queue switching process; see "How the process to switch cluster-sender channel to a different transmission queue works" on page 900.

The process to switch the transmission queue starts when the channel is restarted. The workload rebalancing process starts when the channel is stopped. The two process can run in parallel.

The simple case is when stopping a cluster-sender channel does not cause the rebalancing process to change the cluster-sender channel that is to forward any messages on the queue. This case is when no other cluster-sender channel can forward the messages to the correct destination. With no alternative cluster-sender channel to forward the messages to their destination, the messages remain flagged for the same cluster-sender channel after the cluster-sender channel stops. When the channel starts, if a switch is pending, the switching processes moves the messages to a different transmission queue where they are processed by the same cluster-sender channel.

The more complex case is where more than one cluster-sender channel can process some messages to the same destination. You stop and restart the cluster-sender channel to trigger the transmission queue switch. In many cases, by the time you restart the channel, the workload balancing algorithm has already moved messages from the original transmission queue to different transmission queues served by different cluster-sender channels. Only those messages that cannot be forwarded by a different cluster-sender channel remain to be transferred to the new transmission queue. In some cases, if the channel is restarted quickly, some messages that could be transferred by the workload balancing algorithm remain. In which case some remaining messages are switched by the workload balancing process, and some by the process of switching the transmission queue.

Related concepts:

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

"Calculating the size of the log" on page 1127
Estimating the size of log a queue manager needs.

Related tasks:

"Clustering: Planning how to configure cluster transmission queues" on page 1012
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Adding a queue manager to a cluster: separate transmission queues" on page 926
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

How to choose cluster queue managers to hold full repositories

In each cluster you must choose at least one, and preferably two queue managers to hold full repositories. Two full repositories are sufficient for all but the most exceptional circumstances. If possible, choose queue managers that are hosted on robust and permanently-connected platforms, that do not have coinciding outages, and that are in a central position geographically. Also consider dedicating systems as full repository hosts, and not using these systems for any other tasks.

Full repositories are queue managers that maintain a complete picture of the state of the cluster. To share this information, each full repository is connected by `CLUSSDR` channels (and their corresponding `CLUSRCVR` definitions) to every other full repository in the cluster. You must manually define these channels.

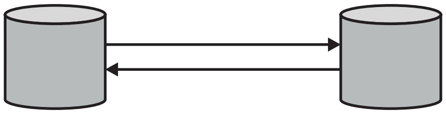


Figure 133. Two connected full repositories.

Every other queue manager in the cluster maintains a picture of what it currently knows about the state of the cluster in a *partial repository*. These queue managers publish information about themselves, and request information about other queue managers, using any two available full repositories. If a chosen full repository is not available, another is used. When the chosen full repository becomes available again, it collects the latest new and changed information from the others so that they keep in step. If all the full repositories go out of service, the other queue managers use the information they have in their partial repositories. However, they are limited to using the information that they have; new information and requests for updates cannot be processed. When the full repositories reconnect to the network, messages are exchanged to bring all repositories (both full and partial) up to date.

When planning the allocation of full repositories, include the following considerations:

- The queue managers chosen to hold full repositories need to be reliable and managed. Choose queue managers that are hosted on a robust and permanently-connected platform.
- Consider the planned outages for the systems hosting your full repositories, and ensure that they do not have coinciding outages.
- Consider network performance: Choose queue managers that are in a central position geographically, or that share the same system as other queue managers in the cluster.
- Consider whether a queue manager is a member of more than one cluster. It can be administratively convenient to use the same queue manager to host the full repositories for several clusters, provided this benefit is balanced against how busy you expect the queue manager to be.
- Consider dedicating some systems to contain only full repositories, and not using these systems for any other tasks. This ensures that these systems only require maintenance for queue manager configuration, and are not removed from service for the maintenance of other business applications. It also ensures that the task of maintaining the repository does not compete with applications for system resources. This can be particularly beneficial in large clusters (say, clusters of more than a thousand queue managers), where full repositories have a much higher workload in maintaining the cluster state.

Having more than two full repositories is possible, but rarely recommended. Although object definitions (that is, queues, topics and channels) flow to all available full repositories, requests only flow from a partial repository to a maximum of two full repositories. This means that, when more than two full repositories are defined, and any two full repositories become unavailable, some partial repositories might not receive updates they would expect. See WMQ Clusters: Why only two Full Repositories?

One situation in which you might find it useful to define more than two full repositories is when migrating existing full repositories to new hardware or new queue managers. In this case, you should introduce the replacement full repositories, and confirm that they have become fully populated, before you remove the previous full repositories. Whenever you add a full repository, remember that you must directly connect it to every other full repository with CLUSSDR channels.

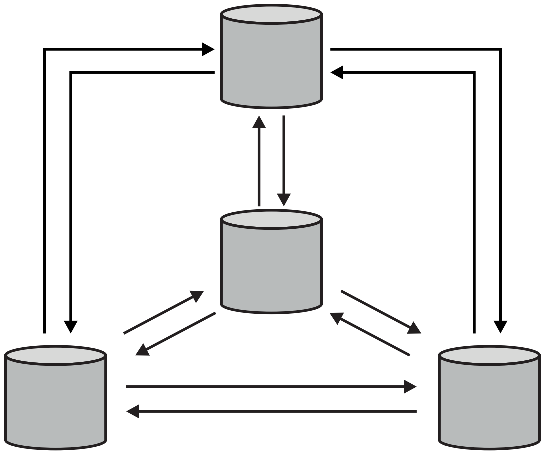


Figure 134. More than two connected full repositories

Related information:

WMQ Clusters: Why only two Full Repositories?

How big can a WMQ Cluster be?

Organizing a cluster

Select which queue managers to link to which full repository. Consider the performance effect, the queue manager version, and whether multiple CLUSSDR channels are desirable.

Having selected the queue managers to hold full repositories, you need to decide which queue managers to link to which full repository. The CLUSSDR channel definition links a queue manager to a full repository from which it finds out about the other full repositories in the cluster. From then on, the queue manager sends messages to any two full repositories. It always tries to use the one to which it has a CLUSSDR channel definition first. You can choose to link a queue manager to either full repository. In choosing, consider the topology of your configuration, and the physical or geographical location of the queue managers.

Because all cluster information is sent to two full repositories, there might be situations in which you want to make a second CLUSSDR channel definition. You might define a second CLUSSDR channel in a cluster that has many full repositories spread over a wide area. You can then control which two full repositories your information is sent to.

Cluster naming conventions

Consider naming queue managers in the same cluster using a naming convention that identifies the cluster to which the queue manager belongs. Use a similar naming convention for channel names and extend it to describe the channel characteristics.

This information contains the old guidance on naming conventions, and the current guidance. As the IBM WebSphere MQ technology improves, and as customers use technology in new or different ways, new recommendations and information must be provided for these scenarios.

Cluster naming conventions: Old best practices

When setting up a new cluster, consider a naming convention for the queue managers. Every queue manager must have a different name. If you give queue managers in a cluster a set of similar names, it might help you to remember which queue managers are grouped where.

When defining channels, remember that all cluster-sender channels have the same name as their corresponding cluster-receiver channel. Channel names are limited to a maximum of 20 characters.

Every cluster-receiver channel must also have a unique name. One possibility is to use the *queue-manager* name preceded by the *cluster-name*. For example, if the *cluster-name* is CLUSTER1 and the *queue-managers* are QM1, QM2, then cluster-receiver channels are CLUSTER1.QM1, CLUSTER1.QM2.

You might extend this convention if channels have different priorities or use different protocols; for example, CLUSTER1.QM1.S1, CLUSTER1.QM1.N3, and CLUSTER1.QM1.T4. In this example, S1 might be the first SNA channel, N3 might be the NetBIOS channel with a network priority of three.

A final qualifier might describe the class of service the channel provides.

The problem with using generic connection names for cluster-receiver definitions is as follows. If you define a CLUSRCVR with a generic CONNAME there is no guarantee that your CLUSSDR channels point to the queue managers you intend. Your initial CLUSSDR might end up pointing to any queue manager in the queue-sharing group, not necessarily one that hosts a full repository. If a channel starts trying a connection again, it might reconnect to a different queue manager with the same generic name disrupting the flow of messages.

Cluster naming conventions: Current best practices

A good naming convention can help to reduce confusion over ownership and scope of clusters. A clear naming convention throughout the cluster topology causes a lot less confusion if clusters get merged at a later time. This situation is also improved if everyone involved is clear about who owns which queue managers and which clusters. Probably the most important point for cluster naming conventions is to put the queue manager name in the channel name, see the following example:

```
CLUSNAME.QMGRNAME
```

This convention might not be obvious to experienced IBM WebSphere MQ users that are unfamiliar with clusters. This oversight is because the XXX.TO.YYY format is such a common method. For example, CLUSTER.TO.XX or CLUSTER.X are commonly used formats that are not recommended for clustering, as they can quickly reach the 20 character limit. The commonly used CLUSTER.TO.XX format becomes confusing if another channel is added later (for example when joining another cluster).

Other objects also benefit from sensible rules, such as: LOB.PROJECT.QNAME or LOB.CLUSTER.ALIAS.NAME.

Queue-sharing groups and clusters

Shared queues can be cluster queues and queue managers in a queue-sharing group can also be cluster queue managers.

Shared queues can also be cluster queues. Furthermore, the queue managers in a queue-sharing group can also be in one or more clusters.

Overlapping clusters

Overlapping clusters provide additional administrative capabilities. Use namelists to reduce the number of commands needed to administer overlapping clusters.

You can create clusters that overlap. There are a number of reasons you might define overlapping clusters; for example:

- To allow different organizations to have their own administration.
- To allow independent applications to be administered separately.
- To create classes of service.

In Figure 135, the queue manager STF2 is a member of both the clusters. When a queue manager is a member of more than one cluster, you can take advantage of namelists to reduce the number of definitions you need. Namelists contain a list of names, for example, cluster names. You can create a namelist naming the clusters. Specify the namelist on the ALTER QMGR command for STF2 to make it a full repository queue manager for both clusters.

If you have more than one cluster in your network, you must give them different names. If two clusters with the same name are ever merged, it is not possible to separate them again. It is also a good idea to give the clusters and channels different names. They are more easily distinguished when you look at the output from the DISPLAY commands. Queue manager names must be unique within a cluster for it to work correctly.

Defining classes of service

Imagine a university that has a queue manager for each member of staff and each student. Messages between members of staff are to travel on channels with a high priority and high bandwidth. Messages between students are to travel on cheaper, slower channels. You can set up this network using traditional distributed queuing techniques. WebSphere MQ selects which channels to use by looking at the destination queue name and queue manager name.

To clearly differentiate between the staff and students, you could group their queue managers into two clusters as shown in Figure 135. WebSphere MQ moves messages to the meetings queue in the staff cluster only over channels that are defined in that cluster. Messages for the gossip queue in the students cluster go over channels defined in that cluster and receive the appropriate class of service.

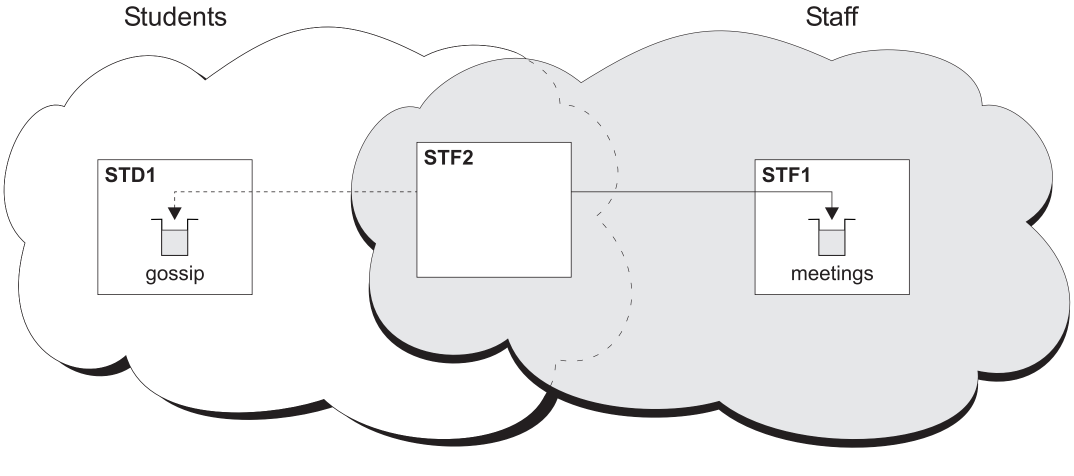


Figure 135. Classes of service

Clustering tips

You might need to make some changes to your systems or applications before using clustering. There are both similarities and differences from the behavior of distributed queuing.

- The WebSphere MQ Explorer cannot directly administer WebSphere MQ for z/OS queue managers at versions earlier than Version 6.0.
- You must add manual configuration definitions to queue managers outside a cluster for them to access cluster queues.
- If you merge two clusters with the same name, you cannot separate them again. Therefore it is advisable to give all clusters a unique name.
- If a message arrives at a queue manager but there is no queue there to receive it, the message is put on the dead-letter queue. If there is no dead-letter queue, the channel fails and tries again. The use of the dead-letter queue is the same as with distributed queuing.
- The integrity of persistent messages is maintained. Messages are not duplicated or lost as a result of using clusters.
- Using clusters reduces system administration. Clusters make it easy to connect larger networks with many more queue managers than you would be able to contemplate using distributed queuing. There is a risk that you might consume excessive network resources if you attempt to enable communication between every queue manager in a cluster.
- If you use the WebSphere MQ Explorer, which presents the queue managers in a tree structure, the view for large clusters might be cumbersome.
- WebSphere MQ Explorer can administer a cluster with repository queue managers on WebSphere MQ for z/OS Version 6 or later. You need not nominate an additional repository on a separate system. For earlier versions of WebSphere MQ on z/OS, the WebSphere MQ Explorer cannot administer a cluster with repository queue managers. You must nominate an additional repository on a system that the WebSphere MQ Explorer can administer.
- The purpose of distribution lists is to use a single MQPUT command to send the same message to multiple destinations. Distribution lists are supported on WebSphere MQ for AIX, IBM i, HP-UX, Solaris, Linux, and Windows. You can use distribution lists with queue manager clusters. In a cluster, all the messages are expanded at MQPUT time. The advantage, in terms of network traffic, is not so great as in a non-clustering environment. The advantage of distribution lists is that the numerous channels and transmission queues do not need to be defined manually.
- If you are going to use clusters to balance your workload examine your applications. See whether they require messages to be processed by a particular queue manager or in a particular sequence. Such applications are said to have message affinities. You might need to modify your applications before you can use them in complex clusters.
- You might choose to use the `MQOO_BIND_ON_OPEN` option on an MQOPEN to force messages to be sent to a specific destination. If the destination queue manager is not available the messages are not delivered until the queue manager becomes available again. Messages are not routed to another queue manager because of the risk of duplication.
- If a queue manager is to host a cluster repository, you need to know its host name or IP address. You have to specify this information in the `CONNAME` parameter when you make the `CLUSSDR` definition on other queue managers joining the cluster. If you use DHCP, the IP address is subject to change because DHCP can allocate a new IP address each time you restart a system. Therefore, you must not specify the IP address in the `CLUSSDR` definitions. Even if all the `CLUSSDR` definitions specify the host name rather than the IP address, the definitions would still not be reliable. DHCP does not necessarily update the DNS directory entry for the host with the new address. If you must nominate queue managers as full repositories on systems that use DHCP, install software that guarantees to keep your DNS directory up to date.
- Do not use generic names, for example VTAM generic resources or Dynamic Domain Name Server (DDNS) generic names as the connection names for your channels. If you do, your channels might connect to a different queue manager than expected.

- You can only get a message from a local cluster queue, but you can put a message to any queue in a cluster. If you open a queue to use the MQGET command, the queue manager opens the local queue.
- You do not need to alter any of your applications if you set up a simple WebSphere MQ cluster. The application can name the target queue on the MQOPEN call and does not need to know about the location of the queue manager. If you set up a cluster for workload management you must review your applications and modify them as necessary.
- You can view current monitoring and status data for a channel or queue using the `DISPLAY CHSTATUS` and the `DISPLAY QSTATUS` **`runmqsc`** commands. The monitoring information can be used to help gauge the performance and health of the system. Monitoring is controlled by queue manager, queue, and channel attributes. Monitoring of auto-defined cluster-sender channels is possible with the `MONACLS` queue manager attribute.

Related concepts:

"Comparison of clustering and distributed queuing" on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

"Components of a cluster" on page 890
Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

"Managing IBM WebSphere MQ clusters" on page 912
You can create, extend, and maintain IBM WebSphere MQ clusters.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

"Setting up a new cluster" on page 913
Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Related information:

Clusters

How clusters work

Establishing communication in a cluster

A channel initiator is needed to start a communication channel when there is a message to deliver. A channel listener waits to start the other end of a channel to receive the message.

Before you begin

To establish communication between queue managers in a cluster, configure a link using one of the supported communication protocols. The supported protocols are TCP or LU 6.2 on any platform, and NetBIOS or SPX on Windows systems. As part of this configuration, you also need channel initiators and channel listeners just as you do with distributed queuing.

About this task

All cluster queue managers need a channel initiator to monitor the system-defined initiation queue `SYSTEM.CHANNEL.INITQ`. `SYSTEM.CHANNEL.INITQ` is the initiation queue for all transmission queues including the cluster transmission queue.

Each queue manager must have a channel listener. A channel listener program waits for incoming network requests and starts the appropriate receiver-channel when it is needed. The implementation of channel listeners is platform-specific, however there are some common features. On all WebSphere MQ platforms, the listener can be started using the `START LISTENER` command. On WebSphere MQ for IBM i, Windows, UNIX and Linux systems, you can start the listener automatically at the same time as the queue manager. To start the listener automatically, set the `CONTROL` attribute of the `LISTENER` object to `QMGR` or `STARTONLY`.

Procedure

1. Start the channel initiator.

   - Windows UNIX Linux

     WebSphere MQ for Windows, UNIX and Linux systems

     When you start a queue manager, if the queue manager attribute `SCHINIT` is set to `QMGR`, a channel initiator is automatically started. Otherwise it can be started using the **runmqsc** `START CHINIT` command or the **runmqchi** control command.

2. Start the channel listener.

   - Windows

     WebSphere MQ for Windows

     Use either the channel listener program provided by WebSphere MQ, or the facilities provided by the operating system.

     To start the WebSphere MQ channel listener use the `RUNMQLSR` command. For example:

     ```
     RUNMQLSR -t tcp -p 1414 -m QM1
     ```

   - UNIX Linux

     WebSphere MQ on UNIX and Linux systems

     Use either the channel listener program provided by WebSphere MQ, or the facilities provided by the operating system; for example, **inetd** for TCP communications.

     To start the WebSphere MQ channel listener use the **runmqlsr** command. For example:

     ```
     runmqlsr -t tcp -p 1414 -m QM1
     ```

     To use **inetd** to start channels, configure two files:

     a. Edit the file `/etc/services`. You must be logged in as a superuser or root. If the following line is not in the file, add it as shown:

        ```
        MQSeries       1414/tcp      # Websphere MQ channel listener
        ```

        where 1414 is the port number required by WebSphere MQ. You can change the port number, but it must match the port number specified at the sending end.

     b. Edit the file `/etc/inetd.conf`. If you do not have the following line in that file, add it as shown:

        ```
        MQSeries stream tcp nowait mqm MQ_INSTALLATION_PATH/bin/amqcrsta amqcrsta
        -m queue.manager.name
        ```

        where *MQ_INSTALLATION_PATH* is replaced by the high-level directory in which WebSphere MQ is installed.

     The updates become active after **inetd** has reread the configuration files. Issue the following commands from the root user ID:

     On AIX:

     ```
     refresh -s inetd
     ```

     On HP-UX:

     ```
     inetd -c
     ```

     On Solaris or Linux:

     a. Find the process ID of the **inetd** with the command:

        ```
        ps -ef | grep inetd
        ```

b. Run the appropriate command, as follows:
   - For Solaris 9 and Linux:

     ```
     kill -1 inetd processid
     ```
   - For Solaris 10, or later versions:

     ```
     inetconv
     ```

How long do the queue manager repositories retain information?

Queue manager repositories retain information for 30 days. An automatic process efficiently refreshes information that is being used.

When a queue manager sends out some information about itself, the full and partial repository queue managers store the information for 30 days. Information is sent out, for example, when a queue manager advertises the creation of a new queue. To prevent this information from expiring, queue managers automatically resend all information about themselves after 27 days. If a partial repository sends a new request for information part way through the 30 day lifetime, the expiry time remains the original 30 days.

When information expires, it is not immediately removed from the repository. Instead it is held for a grace period of 60 days. If no update is received within the grace period, the information is removed. The grace period allows for the fact that a queue manager might have been temporarily out of service at the expiry date. If a queue manager becomes disconnected from a cluster for more than 90 days, it stops being part of the cluster. However, if it reconnects to the network it becomes part of the cluster again. Full repositories do not use information that has expired to satisfy new requests from other queue managers.

Similarly, when a queue manager sends a request for up-to-date information from a full repository, the request lasts for 30 days. After 27 days WebSphere MQ checks the request. If it has been referenced during the 27 days, it is refreshed automatically. If not, it is left to expire and is refreshed by the queue manager if it is needed again. Expiring requests prevents a buildup of requests for information from dormant queue managers.

Note: For large clusters, it can be disruptive if many queue managers automatically resend all information about themselves at the same time. See “Refreshing in a large cluster can affect performance and availability of the cluster” on page 1030.

Related concepts:

“Clustering: Using REFRESH CLUSTER best practices” on page 1030
You use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster. You should not need to use this command, except in exceptional circumstances. If you do need to use it, there are special considerations about how you use it. This information is a guide based on testing and feedback from customers.

Managing IBM WebSphere MQ clusters

You can create, extend, and maintain IBM WebSphere MQ clusters.

For details on how to manage your IBM WebSphere MQ clusters, see the following subtopics:

Related concepts:

"Comparison of clustering and distributed queuing" on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

"Components of a cluster" on page 890
Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

"Setting up a new cluster"
Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Related information:

Clusters

How clusters work

Setting up a new cluster

Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Before you begin

- Instead of following these instructions, you can use one of the wizards supplied with IBM WebSphere MQ Explorer to create a cluster like the one created by this task. Right-click the Queue Manager Clusters folder, then click **New** > **Queue Manager Cluster**, and follow the instructions given in the wizard.
- For background information which might aid your understanding about the steps taken to set up a cluster, see "Cluster queues" on page 892, Channels and Listeners.

About this task

You are setting up a new IBM WebSphere MQ network for a chain store. The store has two branches, one in London and one in New York. The data and applications for each store are hosted by systems running separate queue managers. The two queue managers are called `LONDON` and `NEWYORK`. The inventory application runs on the system in New York, connected to queue manager `NEWYORK`. The application is driven by the arrival of messages on the `INVENTQ` queue, hosted by `NEWYORK`. The two queue managers, `LONDON` and `NEWYORK`, are to be linked in a cluster called `INVENTORY` so that they can both put messages to the `INVENTQ`.

Figure 136 on page 914 shows what this cluster looks like.

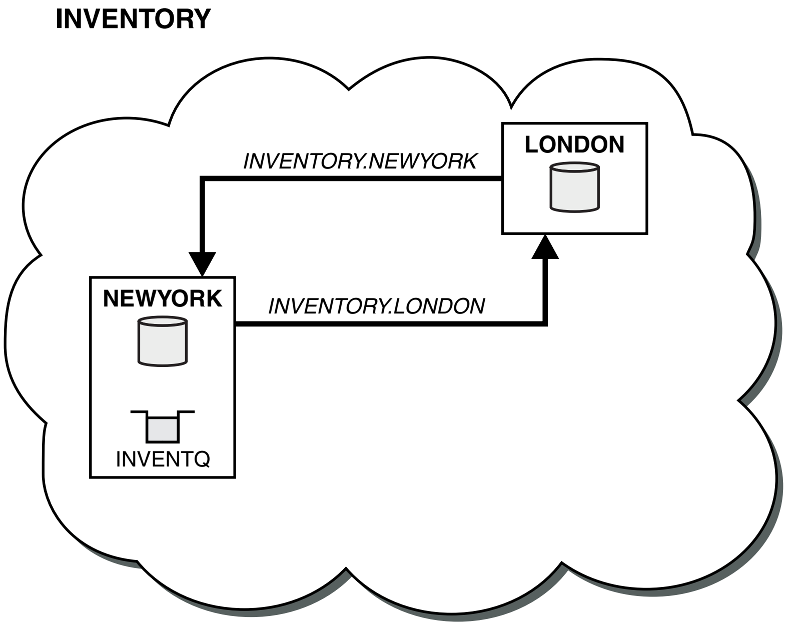


Figure 136. The INVENTORY cluster with two queue managers

You can configure each queue manager in the cluster that is not on z/OS to send messages to other queue managers in the cluster using different cluster transmission queues.

The instructions to set up the cluster vary a little by transport protocol , number of transmission queues, or platform. You have a choice of three combinations. The verification procedure remains the same for all combinations.

Procedure

- “Setting up a cluster using TCP/IP with a single transmission queue per queue manager” on page 915
- “Setting up a cluster on TCP/IP using multiple transmission queues per queue manager” on page 917
- “Setting up a cluster using LU 6.2 on z/OS” on page 920
- “Verifying the cluster” on page 922

Results

Figure 136 shows the `INVENTORY` cluster setup by this task.

Clearly, `INVENTORY` is a small cluster. However, it is useful as a proof of concept. The important thing to understand about this cluster is the scope it offers for future enhancement.

Related concepts:

“Comparison of clustering and distributed queuing” on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

“Components of a cluster” on page 890
Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

“Managing IBM WebSphere MQ clusters” on page 912
You can create, extend, and maintain IBM WebSphere MQ clusters.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

Related information:

Clusters

How clusters work

Setting up a cluster using TCP/IP with a single transmission queue per queue manager:
Before you begin

- On AIX, HP-UX, IBM i, Linux, Solaris, and Windows, the queue manager attribute, **`DEFCLXQ`**, must be left as its default value, `SCTQ`.

About this task

Follow these steps to set up a cluster on AIX, HP-UX, IBM i, Linux, Solaris, and Windows using the transport protocol TCP/IP.

Procedure

1. Decide on the organization of the cluster and its name.

   You decided to link the two queue managers, `LONDON` and `NEWYORK`, into a cluster. A cluster with only two queue managers offers only marginal benefit over a network that is to use distributed queuing. It is a good way to start and it provides scope for future expansion. When you open new branches of your store, you are able to add the new queue managers to the cluster easily. Adding new queue managers does not disrupt the existing network; see "Adding a queue manager to a cluster" on page 924.

   For the time being, the only application you are running is the inventory application. The cluster name is `INVENTORY`.

2. Decide which queue managers are to hold full repositories.

   In any cluster you must nominate at least one queue manager, or preferably two, to hold full repositories. In this example, there are only two queue managers, `LONDON` and `NEWYORK`, both of which hold full repositories.

   a. You can perform the remaining steps in any order.

   b. As you proceed through the steps, warning messages might be written to the queue-manager log. The messages are a result of missing definitions that you have yet to add.

```
Examples of the responses to the commands are shown in a box
like this after each step in this task.
These examples show the responses returned by WebSphere MQ for AIX.
The responses vary on other platforms.
```

   c. Before proceeding with these steps, make sure that the queue managers are started.

3. Alter the queue-manager definitions to add repository definitions.

   On each queue manager that is to hold a full repository, alter the local queue-manager definition, using the `ALTER QMGR` command and specifying the `REPOS` attribute:

   ```
   ALTER QMGR REPOS(INVENTORY)
   ```

For example, if you enter:

```
1 : ALTER QMGR REPOS(INVENTORY)
AMQ8005: Websphere MQ queue manager changed.
```

a. `runmqsc LONDON`
b. `ALTER QMGR REPOS(INVENTORY)`

LONDON is changed to a full repository.

4. Define the listeners.

   Define a listener that accepts network requests from other queue managers for every queue manager in the cluster. On the LONDON queue managers, issue the following command:

   ```
   DEFINE LISTENER(LONDON_LS) TRPTYPE(TCP) CONTROL(QMGR)
   ```

   Note: When you define a listener, a port number should be defined if you are using IP addresses in the CONNAME field and the port number is not the default port (1414). For example:

   ```
   DEFINE LISTENER(LONDON_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(1415)
   ```

   The CONTROL attribute ensures that the listener starts and stops when the queue manager does.

   The listener is not started when it is defined, so it must be manually started the first time with the following MQSC command:

   ```
   START LISTENER(LONDON_LS)
   ```

   Issue similar commands for all the other queue managers in the cluster, changing the listener name for each one.

   There are several ways to define these listeners, as shown in Listeners.

5. Define the CLUSRCVR channel for the LONDON queue manager.

   On every queue manager in a cluster, define a cluster-receiver channel on which the queue manager can receive messages. CLUSRCVR defines the connection name of the queue manager. The connection name is stored in the repositories, where other queue managers can refer to it. The CLUSTER keyword shows the availability of the queue manager to receive messages from other queue managers in the cluster.

   In this example the channel name is INVENTORY.LONDON, and the connection name (CONNAME) is the network address of the machine the queue manager resides on, which is LONDON.CHSTORE.COM. The network address can be entered as an alphanumeric DNS host name, or an IP address in either in IPv4 dotted decimal form. For example, 192.0.2.0, or IPv6 hexadecimal form; for example 2001:DB8:0204:acff:fe97:2c34:fde0:3485. The port number is not specified, so the default port (1414) is used.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-receiver channel for queue manager LONDON')
   ```

   ```
   1 : DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-receiver channel for queue manager LONDON')
   AMQ8014: Websphere MQ channel created.
   07/09/98  12:56:35  No repositories for cluster 'INVENTORY'
   ```

6. Define the CLUSRCVR channel for the NEWYORK queue manager. If the channel listener is using the default port, typically 1414, and the cluster does not include a queue manager on z/OS, you can omit the CONNAME

```
DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CLUSTER(INVENTORY)
DESCR('TCP Cluster-receiver channel for queue manager NEWYORK')
```

7. Define the CLUSSDR channel on the LONDON queue manager.

   On every queue manager in a cluster, define one cluster-sender channel. The queue manager sends messages to one of the full repository queue managers on the cluster-sender channel. In this case, there are only two queue managers, both of which hold full repositories. They must each have a CLUSSDR definition that points to the CLUSRCVR channel defined at the other queue manager. The channel names given on the CLUSSDR definitions must match the channel names on the corresponding CLUSRCVR definitions. Once a queue manager has definitions for both a cluster-receiver channel and a cluster-sender channel in the same cluster, the cluster-sender channel is started.

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-sender channel from LONDON to repository at NEWYORK')
   ```

```
1 : DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
DESCR('TCP Cluster-sender channel from LONDON to repository at NEWYORK')
AMQ8014: Websphere MQ channel created.
07/09/98  13:00:18    Channel program started.
```

8. Define the CLUSSDR channel on the NEWYORK queue manager.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-sender channel from NEWYORK to repository at LONDON')
   ```

9. Define the cluster queue INVENTQ

   Define the INVENTQ queue on the NEWYORK queue manager, specifying the CLUSTER keyword.

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
   ```

```
1 : DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
AMQ8006: Websphere MQ queue created.
```

   The CLUSTER keyword causes the queue to be advertised to the cluster. As soon as the queue is defined it becomes available to the other queue managers in the cluster. They can send messages to it without having to make a remote-queue definition for it.

   All the definitions are complete. On all platforms, start a listener program on each queue manager. The listener program waits for incoming network requests and starts the cluster-receiver channel when it is needed.

Setting up a cluster on TCP/IP using multiple transmission queues per queue manager:
About this task

Follow these steps to set up a cluster on AIX, HP-UX, IBM i, Linux, Solaris, and Windows using the transport protocol TCP/IP. The repository queue managers are configured to use a different cluster transmission queue to send messages to each other, and to other queue managers in the cluster. If you add queue managers to the cluster that are also to use different transmission queues, follow the task, “Adding a queue manager to a cluster: separate transmission queues” on page 926. You cannot set up a queue manager on z/OS to use separate cluster transmission queues.

Procedure

1. Decide on the organization of the cluster and its name.

You decided to link the two queue managers, LONDON and NEWYORK, into a cluster. A cluster with only two queue managers offers only marginal benefit over a network that is to use distributed queuing. It is a good way to start and it provides scope for future expansion. When you open new branches of your store, you are able to add the new queue managers to the cluster easily. Adding new queue managers does not disrupt the existing network; see “Adding a queue manager to a cluster” on page 924.

For the time being, the only application you are running is the inventory application. The cluster name is INVENTORY.

2. Decide which queue managers are to hold full repositories.

   In any cluster you must nominate at least one queue manager, or preferably two, to hold full repositories. In this example, there are only two queue managers, LONDON and NEWYORK, both of which hold full repositories.

   a. You can perform the remaining steps in any order.

   b. As you proceed through the steps, warning messages might be written to the queue-manager log. The messages are a result of missing definitions that you have yet to add.

```
Examples of the responses to the commands are shown in a box
like this after each step in this task.
These examples show the responses returned by WebSphere MQ for AIX.
The responses vary on other platforms.
```

   c. Before proceeding with these steps, make sure that the queue managers are started.

3. Alter the queue-manager definitions to add repository definitions.

   On each queue manager that is to hold a full repository, alter the local queue-manager definition, using the ALTER QMGR command and specifying the REPOS attribute:

   ```
   ALTER QMGR REPOS(INVENTORY)
   ```

   For example, if you enter:

```
1 : ALTER QMGR REPOS(INVENTORY)
AMQ8005: Websphere MQ queue manager changed.
```

   a. runmqsc LONDON

   b. ALTER QMGR REPOS(INVENTORY)

   LONDON is changed to a full repository.

4. Alter the queue-manager definitions to create separate cluster transmission queues for each destination.

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

   On each queue manager that you add to the cluster decide whether to use separate transmission queues or not. See the topics, “Adding a queue manager to a cluster” on page 924 and “Adding a queue manager to a cluster: separate transmission queues” on page 926.

5. Define the listeners.

   Define a listener that accepts network requests from other queue managers for every queue manager in the cluster. On the LONDON queue managers, issue the following command:

   ```
   DEFINE LISTENER(LONDON_LS) TRPTYPE(TCP) CONTROL(QMGR)
   ```

   Note: When you define a listener, a port number should be defined if you are using IP addresses in the CONNAME field and the port number is not the default port (1414). For example:

   ```
   DEFINE LISTENER(LONDON_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(1415)
   ```

   The CONTROL attribute ensures that the listener starts and stops when the queue manager does.

The listener is not started when it is defined, so it must be manually started the first time with the following MQSC command:

```
START LISTENER(LONDON_LS)
```

Issue similar commands for all the other queue managers in the cluster, changing the listener name for each one.

There are several ways to define these listeners, as shown in Listeners.

6. Define the CLUSRCVR channel for the `LONDON` queue manager.

   On every queue manager in a cluster, define a cluster-receiver channel on which the queue manager can receive messages. `CLUSRCVR` defines the connection name of the queue manager. The connection name is stored in the repositories, where other queue managers can refer to it. The `CLUSTER` keyword shows the availability of the queue manager to receive messages from other queue managers in the cluster.

   In this example the channel name is `INVENTORY.LONDON`, and the connection name (`CONNAME`) is the network address of the machine the queue manager resides on, which is `LONDON.CHSTORE.COM`. The network address can be entered as an alphanumeric DNS host name, or an IP address in either in IPv4 dotted decimal form. For example, `192.0.2.0`, or IPv6 hexadecimal form; for example `2001:DB8:0204:acff:fe97:2c34:fde0:3485`. The port number is not specified, so the default port (1414) is used.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-receiver channel for queue manager LONDON')
   ```

```
1 : DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
DESCR('TCP Cluster-receiver channel for queue manager LONDON')
AMQ8014: Websphere MQ channel created.
07/09/98  12:56:35  No repositories for cluster 'INVENTORY'
```

7. Define the CLUSRCVR channel for the `NEWYORK` queue manager. If the channel listener is using the default port, typically 1414, and the cluster does not include a queue manager on z/OS, you can omit the `CONNAME`

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-receiver channel for queue manager NEWYORK')
   ```

8. Define the CLUSSDR channel on the `LONDON` queue manager.

   On every queue manager in a cluster, define one cluster-sender channel. The queue manager sends messages to one of the full repository queue managers on the cluster-sender channel. In this case, there are only two queue managers, both of which hold full repositories. They must each have a `CLUSSDR` definition that points to the `CLUSRCVR` channel defined at the other queue manager. The channel names given on the `CLUSSDR` definitions must match the channel names on the corresponding `CLUSRCVR` definitions. Once a queue manager has definitions for both a cluster-receiver channel and a cluster-sender channel in the same cluster, the cluster-sender channel is started.

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-sender channel from LONDON to repository at NEWYORK')
   ```

```
1 : DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
DESCR('TCP Cluster-sender channel from LONDON to repository at NEWYORK')
AMQ8014: Websphere MQ channel created.
07/09/98  13:00:18    Channel program started.
```

9. Define the CLUSSDR channel on the NEWYORK queue manager.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('TCP Cluster-sender channel from NEWYORK to repository at LONDON')
   ```

10. Define the cluster queue INVENTQ

    Define the INVENTQ queue on the NEWYORK queue manager, specifying the CLUSTER keyword.

    ```
    DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
    ```

```
1 : DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
AMQ8006: Websphere MQ queue created.
```

The CLUSTER keyword causes the queue to be advertised to the cluster. As soon as the queue is defined it becomes available to the other queue managers in the cluster. They can send messages to it without having to make a remote-queue definition for it.

All the definitions are complete. On all platforms, start a listener program on each queue manager. The listener program waits for incoming network requests and starts the cluster-receiver channel when it is needed.

Setting up a cluster using LU 6.2 on z/OS:
Procedure

1. Decide on the organization of the cluster and its name.

   You decided to link the two queue managers, LONDON and NEWYORK, into a cluster. A cluster with only two queue managers offers only marginal benefit over a network that is to use distributed queuing. It is a good way to start and it provides scope for future expansion. When you open new branches of your store, you are able to add the new queue managers to the cluster easily. Adding new queue managers does not disrupt the existing network; see "Adding a queue manager to a cluster" on page 924.

   For the time being, the only application you are running is the inventory application. The cluster name is INVENTORY.

2. Decide which queue managers are to hold full repositories.

   In any cluster you must nominate at least one queue manager, or preferably two, to hold full repositories. In this example, there are only two queue managers, LONDON and NEWYORK, both of which hold full repositories.

   a. You can perform the remaining steps in any order.
   b. As you proceed through the steps, warning messages might be written the z/OS system console. The messages are a result of missing definitions that you have yet to add.
   c. Before proceeding with these steps, make sure that the queue managers are started.

3. Alter the queue-manager definitions to add repository definitions.

   On each queue manager that is to hold a full repository, alter the local queue-manager definition, using the ALTER QMGR command and specifying the REPOS attribute:

   ```
   ALTER QMGR REPOS(INVENTORY)
   ```

For example, if you enter:

```
1 : ALTER QMGR REPOS(INVENTORY)
AMQ8005: Websphere MQ queue manager changed.
```

a. `runmqsc LONDON`

b. `ALTER QMGR REPOS(INVENTORY)`

`LONDON` is changed to a full repository.

4. Define the listeners.

   The listener is not started when it is defined, so it must be manually started the first time with the following MQSC command:

   ```
   START LISTENER(LONDON_LS)
   ```

   Issue similar commands for all the other queue managers in the cluster, changing the listener name for each one.

5. Define the CLUSRCVR channel for the `LONDON` queue manager.

   On every queue manager in a cluster, define a cluster-receiver channel on which the queue manager can receive messages. `CLUSRCVR` defines the connection name of the queue manager. The connection name is stored in the repositories, where other queue managers can refer to it. The `CLUSTER` keyword shows the availability of the queue manager to receive messages from other queue managers in the cluster.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) TRPTYPE(LU62)
   CONNAME(LONDON.LUNAME) CLUSTER(INVENTORY)
   MODENAME('#INTER') TPNAME('MQSERIES')
   DESCR('LU62 Cluster-receiver channel for queue manager LONDON')
   ```

   ```
   1 : DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) TRPTYPE(LU62)
   CONNAME(LONDON.LUNAME) CLUSTER(INVENTORY)
   MODENAME('#INTER') TPNAME('MQSERIES')
   DESCR('LU62 Cluster-receiver channel for queue manager LONDON')
   AMQ8014: Websphere MQ channel created.
   07/09/98  12:56:35  No repositories for cluster 'INVENTORY'
   ```

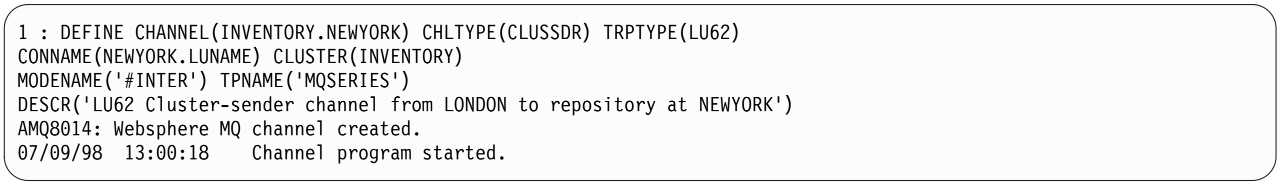
6. Define the CLUSRCVR channel for the `NEWYORK` queue manager.

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSRCVR) TRPTYPE(LU62)
   CONNAME(NEWYORK.LUNAME) CLUSTER(INVENTORY)
   MODENAME('#INTER') TPNAME('MQSERIES')
   DESCR('LU62 Cluster-receiver channel for queue manager NEWYORK')
   ```

7. Define the CLUSSDR channel on the `LONDON` queue manager.

   On every queue manager in a cluster, define one cluster-sender channel. The queue manager sends messages to one of the full repository queue managers on the cluster-sender channel. In this case, there are only two queue managers, both of which hold full repositories. They must each have a `CLUSSDR` definition that points to the `CLUSRCVR` channel defined at the other queue manager. The channel names given on the `CLUSSDR` definitions must match the channel names on the corresponding `CLUSRCVR` definitions. Once a queue manager has definitions for both a cluster-receiver channel and a cluster-sender channel in the same cluster, the cluster-sender channel is started.

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(LU62)
   CONNAME(CPIC) CLUSTER(INVENTORY)
   DESCR('LU62 Cluster-sender channel from LONDON to repository at NEWYORK')
   ```



```
1 : DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(LU62)
CONNAME(NEWYORK.LUNAME) CLUSTER(INVENTORY)
MODENAME('#INTER') TPNAME('MQSERIES')
DESCR('LU62 Cluster-sender channel from LONDON to repository at NEWYORK')
AMQ8014: Websphere MQ channel created.
07/09/98  13:00:18    Channel program started.
```

8. Define the CLUSSDR channel on the NEWYORK queue manager.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSSDR) TRPTYPE(LU62)
   CONNAME(LONDON.LUNAME) CLUSTER(INVENTORY)
   DESCR('LU62 Cluster-sender channel from NEWYORK to repository at LONDON')
   ```

9. Define the cluster queue INVENTQ

   Define the INVENTQ queue on the NEWYORK queue manager, specifying the CLUSTER keyword.

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
   ```

```
1 : DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
AMQ8006: Websphere MQ queue created.
```

The CLUSTER keyword causes the queue to be advertised to the cluster. As soon as the queue is defined it becomes available to the other queue managers in the cluster. They can send messages to it without having to make a remote-queue definition for it.

All the definitions are complete. On all platforms, start a listener program on each queue manager. The listener program waits for incoming network requests and starts the cluster-receiver channel when it is needed.

Verifying the cluster:
About this task

You can verify the cluster in one or more of these ways:

1. Running administrative commands to display cluster and channel attributes.
2. Run the sample programs to send and receive messages on a cluster queue.
3. Write your own programs to send a request message to a cluster queue and reply with a response messages to an non-clustered reply queue.

Procedure

1. Issue DISPLAY **runmqsc** commands to verify the cluster. The responses you see ought to be like the responses in the steps that follow.
   a. From the NEWYORK queue manager, run the **DISPLAY CLUSQMGR** command:

      ```
      dis clusqmgr(*)
      ```

```
1 : dis clusqmgr(*)
AMQ8441: Display Cluster Queue Manager details.
CLUSQMGR(NEWYORK)             CLUSTER(INVENTORY)
CHANNEL(INVENTORY.NEWYORK)
AMQ8441: Display Cluster Queue Manager details.
CLUSQMGR(LONDON)              CLUSTER(INVENTORY)
CHANNEL(INVENTORY.LONDON)
```

b. From the NEWYORK queue manager, run the **DISPLAY CHANNEL STATUS** command:

```
dis chstatus(*)
```

```
1 : dis chstatus(*)
AMQ8417: Display Channel Status details.
CHANNEL(INVENTORY.NEWYORK)    XMITQ( )
CONNAME(192.0.2.0)            CURRENT
CHLTYPE(CLUSRCVR)             STATUS(RUNNING)
RQMNAME(LONDON)
AMQ8417: Display Channel Status details.
CHANNEL(INVENTORY.LONDON)     XMITQ(SYSTEM.CLUSTER.TRANSMIT.INVENTORY.LONDON)
CONNAME(192.0.2.1)            CURRENT
CHLTYPE(CLUSSDR)              STATUS(RUNNING)
RQMNAME(LONDON)
```

2. Send messages between the two queue managers, using **amqsput**.
   a. On LONDON run the command **amqsput INVENTQ LONDON**.

      Type some messages, followed by a blank line.
   b. On NEWYORK run the command **amqsget INVENTQ NEWYORK**.

      You now see the messages you entered on LONDON. After 15 seconds the program ends.
3. Send messages between the two queue managers using your own programs. In the following steps, LONDON puts a message to the INVENTQ at NEWYORK and receives a reply on its queue LONDON_reply.
   a. On LONDON put a messages to the cluster queue.
      1) Define a local queue called LONDON_reply
      2) Set the MQOPEN options to MQOO_OUTPUT
      3) Issue the MQOPEN call to open the queue INVENTQ
      4) Set the *ReplyToQ* name in the message descriptor to LONDON_reply
      5) Issue the MQPUT call to put the message
      6) Commit the message
   b. On NEWYORK receive the message on the cluster queue and put a reply to the reply queue.
      1) Set the MQOPEN options to MQOO_BROWSE
      2) Issue the MQOPEN call to open the queue INVENTQ
      3) Issue the MQGET call to get the message from INVENTQ
      4) Retrieve the *ReplyToQ* name from the message descriptor
      5) Put the *ReplyToQ* name in the ObjectName field of the object descriptor
      6) Set the MQOPEN options to MQOO_OUTPUT
      7) Issue the MQOPEN call to open LONDON_reply at queue manager LONDON
      8) Issue the MQPUT call to put the message to LONDON_reply
   c. On LONDON receive the reply.
      1) Set the MQOPEN options to MQOO_BROWSE
      2) Issue the MQOPEN call to open the queue LONDON_reply

3) Issue the MQGET call to get the message from LONDON_reply

Adding a queue manager to a cluster

Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using the single cluster transmission queue SYSTEM.CLUSTER.TRANSMIT.QUEUE.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster is set up as described in “Setting up a new cluster” on page 913. It contains two queue managers, LONDON and NEWYORK, which both hold full repositories.
- The queue manager PARIS is owned by the primary installation. If it is not, you must run the **setmqenv** command to set up the command environment for the installation that PARIS belongs to.
- TCP connectivity exists between all three systems, and the queue manager is configured with a TCP listener that starts under the control of the queue manager.

About this task

1. A new branch of the chain store is being set up in Paris and you want to add a queue manager called PARIS to the cluster.
2. Queue manager PARIS sends inventory updates to the application running on the system in New York by putting messages on the INVENTQ queue.

Follow these steps to add a queue manager to a cluster.

Procedure

1. Decide which full repository PARIS refers to first.

   Every queue manager in a cluster must refer to one or other of the full repositories. It gathers information about the cluster from a full repository and so builds up its own partial repository. Choose either of the repositories as the full repository. As soon as a new queue manager is added to the cluster it immediately learns about the other repository as well. Information about changes to a queue manager is sent directly to two repositories. In this example, you link PARIS to the queue manager LONDON, purely for geographical reasons.

   Note: Perform the remaining steps in any order, after queue manager PARIS is started.
2. Define a CLUSRCVR channel on queue manager PARIS.

   Every queue manager in a cluster must define a cluster-receiver channel on which it can receive messages. On PARIS, define:

   ```
   DEFINE CHANNEL(INVENTORY.PARIS) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(PARIS.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-receiver channel for queue manager PARIS')
   ```

   The cluster-receiver channel advertises the availability of the queue manager to receive messages from other queue managers in the cluster INVENTORY. There is no need to make definitions on other queue managers for a sending end to the cluster-receiver channel INVENTORY.PARIS. Other definitions are made automatically when needed.
3. Define a CLUSSDR channel on queue manager PARIS.

   Every queue manager in a cluster must define one cluster-sender channel on which it can send messages to its initial full repository. On PARIS, make the following definition for a channel called INVENTORY.LONDON to the queue manager with the network address LONDON.CHSTORE.COM.

```
DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
DESCR('Cluster-sender channel from PARIS to repository at LONDON')
```

4. Optional: If this queue manager is rejoining a cluster, complete some extra steps.
   a. If you are adding a queue manager to a cluster that has previously been removed from the same cluster, check that it is now showing as a cluster member. If not, complete the following extra steps:
      1) Issue the **REFRESH CLUSTER** command on the queue manager you are adding. This step stops the cluster channels, and gives your local cluster cache a fresh set of sequence numbers that are assured to be up-to-date within the rest of the cluster.

         ```
         REFRESH CLUSTER(INVENTORY) REPOS(YES)
         ```

         Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.
      2) Restart the CLUSSDR channel (for example, using the START CHANNEL command).
      3) Restart the CLUSRCVR channel.
   b. If the cluster is a publish/subscribe cluster, and the rejoining queue manager has subscriptions, issue the following command to ensure the proxy subscriptions are correctly synchronized across the cluster:

      ```
      REFRESH QMGR TYPE(PROXYSUB)
      ```

Results

The following figure shows the cluster set up by this task.

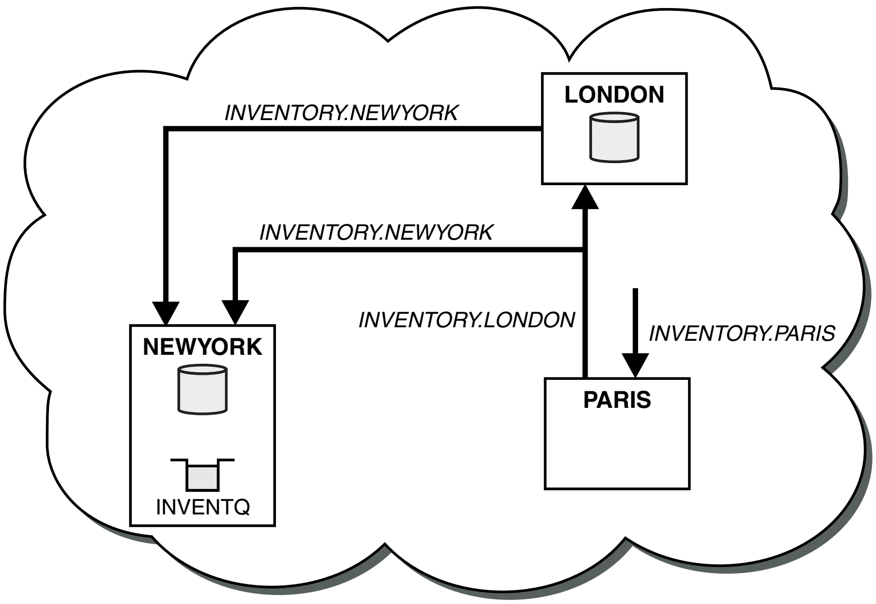


Figure 137. The `INVENTORY` cluster with three queue managers

By making only two definitions, a CLUSRCVR definition and a CLUSSDR definition, we added the queue manager PARIS to the cluster.

Now the PARIS queue manager learns, from the full repository at LONDON, that the INVENTQ queue is hosted by queue manager NEWYORK. When an application hosted by the system in Paris tries to put messages to the INVENTQ, PARIS automatically defines a cluster-sender channel to connect to the cluster-receiver channel INVENTORY.NEWYORK. The application can receive responses when its queue-manager name is specified as the target queue manager and a reply-to queue is provided.

Adding a queue manager to a cluster: separate transmission queues

Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

Before you begin

- The queue manager is defined on a platform other than z/OS.
- The queue manager is not a member of any clusters.
- The cluster exists; there is a full repository to which this queue manager can connect directly and the repository is available. For the steps to create the cluster, see “Setting up a new cluster” on page 913.

About this task

This task is an alternative to “Adding a queue manager to a cluster” on page 924, in which you add a queue manager to a cluster that places cluster messages on a single transmission queue.

In this task, you add a queue manager to a cluster that automatically creates separate cluster transmission queues for each cluster-sender channel.

To keep the number of definitions of queues small, the default is to use a single transmission queue. Using separate transmission queues is advantageous if you want to monitor traffic destined to different queue managers and different clusters. You might also want to separate traffic to different destinations to achieve isolation or performance goals.

Procedure

1. Alter the default cluster channel transmission queue type.

   Alter the queue manager PARIS:

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

   Every time the queue manager creates a cluster-sender channel to send a message to a queue manager, it creates a cluster transmission queue. The transmission queue is used only by this cluster-sender channel. The transmission queue is permanent-dynamic. It is created from the model queue, SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE, with the name SYSTEM.CLUSTER.TRANSMIT.*ChannelName*.
2. Decide which full repository PARIS refers to first.

   Every queue manager in a cluster must refer to one or other of the full repositories. It gathers information about the cluster from a full repository and so builds up its own partial repository. Choose either of the repositories as the full repository. As soon as a new queue manager is added to the cluster it immediately learns about the other repository as well. Information about changes to a queue manager is sent directly to two repositories. In this example, you link PARIS to the queue manager LONDON, purely for geographical reasons.

   Note: Perform the remaining steps in any order, after queue manager PARIS is started.
3. Define a CLUSRCVR channel on queue manager PARIS.

   Every queue manager in a cluster must define a cluster-receiver channel on which it can receive messages. On PARIS, define:

   ```
   DEFINE CHANNEL(INVENTORY.PARIS) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(PARIS.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-receiver channel for queue manager PARIS')
   ```

The cluster-receiver channel advertises the availability of the queue manager to receive messages from other queue managers in the cluster INVENTORY. There is no need to make definitions on other queue managers for a sending end to the cluster-receiver channel INVENTORY.PARIS. Other definitions are made automatically when needed.

4. Define a CLUSSDR channel on queue manager PARIS.

   Every queue manager in a cluster must define one cluster-sender channel on which it can send messages to its initial full repository. On PARIS, make the following definition for a channel called INVENTORY.LONDON to the queue manager with the network address LONDON.CHSTORE.COM.

   ```
   DEFINE CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-sender channel from PARIS to repository at LONDON')
   ```

   The queue manager automatically creates the permanent dynamic cluster transmission queue SYSTEM.CLUSTER.TRANSMIT.INVENTORY.LONDON from the model queue SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE. It sets the CLCHNAME attribute of the transmission queue to INVENTORY.LONDON.

Results

The following figure shows the cluster set up by this task.

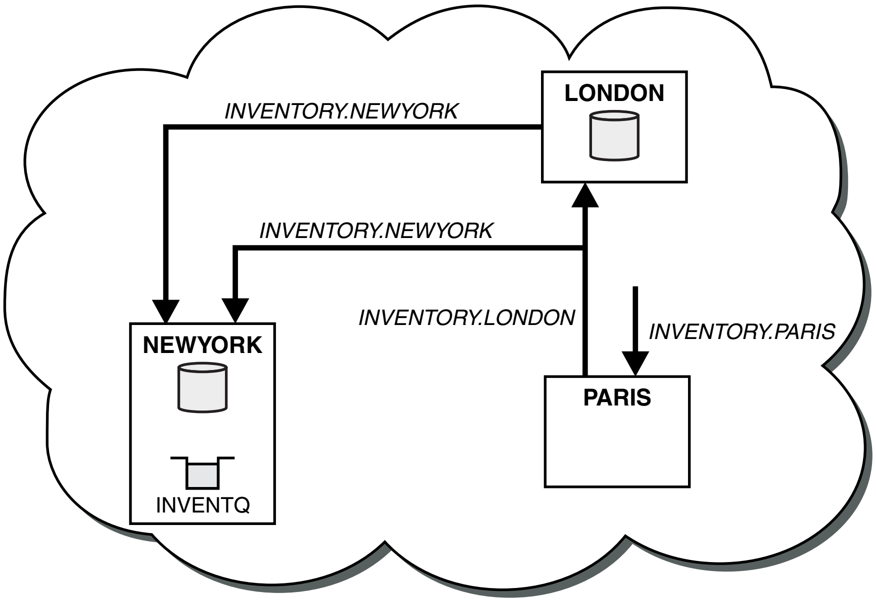


Figure 138. The INVENTORY cluster with three queue managers

By making only two definitions, a CLUSRCVR definition and a CLUSSDR definition, we added the queue manager PARIS to the cluster.

Now the PARIS queue manager learns, from the full repository at LONDON, that the INVENTQ queue is hosted by queue manager NEWYORK. When an application hosted by the system in Paris tries to put messages to the INVENTQ, PARIS automatically defines a cluster-sender channel to connect to the cluster-receiver channel INVENTORY.NEWYORK. The application can receive responses when its queue-manager name is specified as the target queue manager and a reply-to queue is provided.

Adding a remote queue definition to isolate messages sent from a gateway queue manager

Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

Before you begin

Construct the overlapping clusters shown in Figure 144 on page 946 in "Creating two-overlapping clusters with a gateway queue manager" on page 946 by following the steps in that task.

About this task

The solution uses distributed queueing to separate the messages for the `Server App` application from other message traffic on the gateway queue manager. You must define a clustered remote queue definition on `QM1` to divert the messages to a different transmission queue, and a different channel. The remote queue definition must include a reference to the specific transmission queue that stores messages only for `Q1` on `QM3`. In Figure 139 on page 929, the cluster queue alias `Q1A` is supplemented by a remote queue definition `Q1R`, and a transmission queue and sender-channel added.

In this solution, any reply messages are returned using the common `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

The advantage of this solution is that it is easy to separate traffic for multiple destination queues on the same queue manager, in the same cluster. The disadvantage of the solution is that you cannot use cluster workload balancing between multiple copies of `Q1` on different queue managers. To overcome this disadvantage, see "Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931. You also have to manage the switch from one transmission queue to the other.

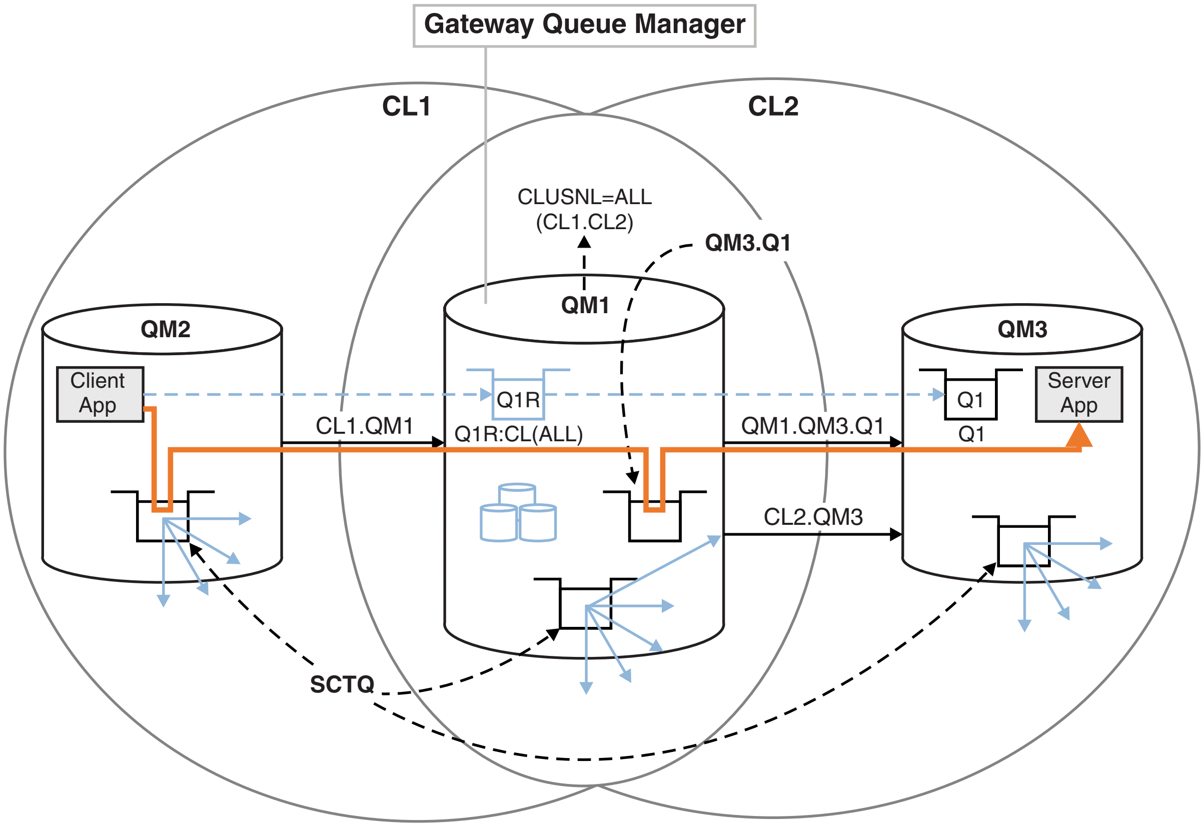


Figure 139. Client-server application deployed to hub and spoke cluster architecture using remote queue definitions

Procedure

1. Create a channel to separate the message traffic for Q1 from the gateway queue manager
   a. Create a sender channel on the gateway queue manager, QM1, to the target queue manager, QM3.
      ```
      DEFINE CHANNEL(QM1.QM3.Q1) CHLTYPE(SDR) CONNAME(QM3HostName(1413)) XMITQ(QM3.Q1) REPLACE
      ```
   b. Create a receiver channel on the target queue manager, QM3.
      ```
      DEFINE CHANNEL(QM1.QM3.Q1) CHLTYPE(RCVR) REPLACE
      ```
2. Create a transmission queue on the gateway queue manager for message traffic to Q1
   ```
   DEFINE QLOCAL(QM3.Q1) USAGE(XMITQ) REPLACE
   START CHANNEL(QM1.QM3.Q1)
   ```
   Starting the channel that is associated with the transmission queue, associates the transmission queue with the channel. The channel starts automatically, once the transmission queue has been associated with the channel.
3. Supplement the clustered queue alias definition for Q1 on the gateway queue manager with a clustered remote queue definition.
   ```
   DEFINE QREMOTE CLUSNL(ALL) RNAME(Q1) RQMNAME(QM3) XMITQ(QM3.Q1) REPLACE
   ```

What to do next

Test the configuration by sending a message to Q1 on QM3 from QM2 using the clustered queue remote definition Q1R on the gateway queue manager QM1.

1. Run the sample program **amqsput** on QM2 to put a message.

```
C:\IBM\MQ>amqsput Q1R QM2
Sample AMQSPUT0 start
```

```
target queue is Q1R
Sample request message from QM2 to Q1 using Q1R

Sample AMQSPUT0 end
```

2. Run the sample program **amqsget** to get the message from Q1 on QM3

   ```
   C:\IBM\MQ>amqsget Q1 QM3
   Sample AMQSGET0 start
   message <Sample request message from QM2 to Q1 using Q1R>
   no more messages
   Sample AMQSGET0 end
   ```

Related concepts:

"Access control and multiple cluster transmission queues" on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against SYSTEM.CLUSTER.TRANSMIT.QUEUE, or checking against local profiles for the cluster queue, or cluster queue manager.

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

Related tasks:

"Adding a queue manager to a cluster: separate transmission queues" on page 926
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

"Changing the default to separate cluster transmission queues to isolate message traffic" on page 951
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

"Clustering: Planning how to configure cluster transmission queues" on page 1012
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager

Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

Before you begin

1. The gateway queue manager must be on Version 7.5, or later, and on a platform other than z/OS.
2. Construct the overlapping clusters shown in Figure 144 on page 946 in "Creating two-overlapping clusters with a gateway queue manager" on page 946 by following the steps in that task.

About this task

On the gateway queue manager, QM1, add a transmission queue and set its queue attribute CLCHNAME. Set CLCHNAME to the name of the cluster-receiver channel on QM3; see Figure 140 on page 932.

This solution has a number of advantages over the solution described in "Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928:

- It requires fewer additional definitions.
- It supports workload balancing between multiple copies of the target queue, Q1, on different queue managers in the same cluster, CL2.
- The gateway queue manager switches automatically to the new configuration when the channel restarts without loosing any messages.
- The gateway queue manager continues to forward messages in the same order as it received them. It does so, even if the switch takes place with messages for the queue Q1 at QM3 still on SYSTEM.CLUSTER.TRANSMIT.QUEUE.

The configuration to isolate cluster message traffic in Figure 140 on page 932 does not result in as great an isolation of traffic as the configuration using remote queues in "Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928. If the queue manager QM3 in CL2 is hosting a number of different cluster queues and server applications, all those queues share the cluster channel, CL2.QM3, connecting QM1 to QM3. The additional flows are illustrated in Figure 140 on page 932 by the gray arrow representing potential cluster message traffic from the SYSTEM.CLUSTER.TRANSMIT.QUEUE to the cluster-sender channel CL2.QM3.

The remedy is to restrict the queue manager to hosting one cluster queue in a particular cluster. If the queue manager is already hosting a number of cluster queues, then to meet this restriction, you must either create another queue manager, or create another cluster; see "Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934.

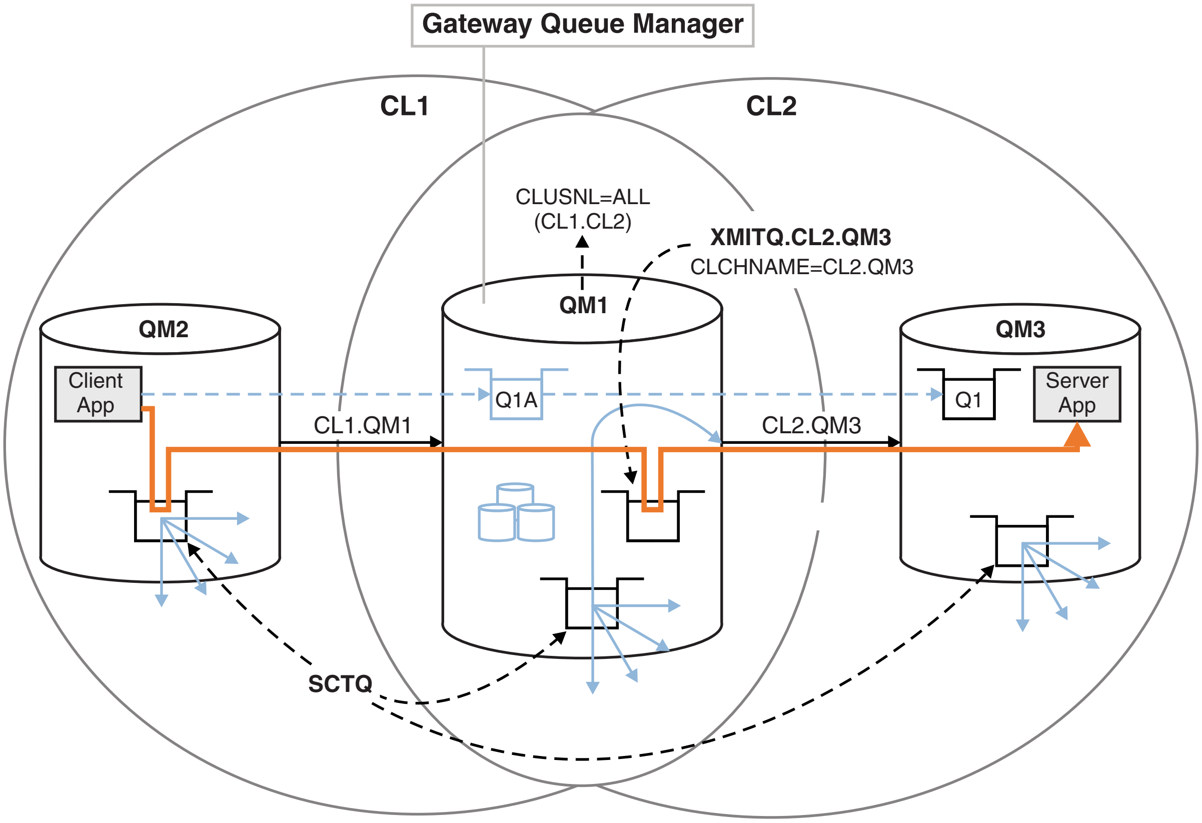


Figure 140. Client-server application deployed to hub and spoke architecture using an additional cluster transmission queue.

Procedure

1. Create an additional cluster transmission queue for the cluster-sender channel CL2.QM3 on the gateway queue manager, QM1.

```
*... on QM1
DEFINE QLOCAL(XMITQ.CL2.QM3) USAGE(XMITQ) CLCHNAME(CL2.QM3)
```

2. Switch to using the transmission queue, XMITQ.CL2.QM3.

   a. Stop the cluster-sender channel CL2.QM3.

```
*... On QM1
STOP CHANNEL(CL2.QM3)
```

   The response is that the command is accepted:

```
AMQ8019: Stop WebSphere MQ channel accepted.
```

   b. Check that the channel CL2.QM3 is stopped

   If the channel does not stop, you can run the **STOP CHANNEL** command again with the FORCE option. An example of setting the FORCE option would be if the channel does not stop, and you cannot restart the other queue manager to synchronize the channel.

```
*... On QM1
start
```

   The response is a summary of the channel status

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM3)              CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1413))      CURRENT
   RQMNAME(QM3)                 STATUS(STOPPED)
   SUBSTATE(MQGET)              XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
```

c. Start the channel, CL2.QM3.

```
*... On QM1
START CHANNEL(CL2.QM3)
```

The response is that the command is accepted:

```
AMQ8018: Start WebSphere MQ channel accepted.
```

d. Check the channel started.

```
*... On QM1
DISPLAY CHSTATUS(CL2.QM3)
```

The response is a summary of the channel status:

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM3)                   CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1413))           CURRENT
   RQMNAME(QM3)                       STATUS(RUNNING)
   SUBSTATE(MQGET)                    XMITQ(XMITQ.CL2.QM3)
```

e. Check the transmission queue was switched.

Monitor the gateway queue manager error log for the message “AMQ7341 The transmission queue for channel CL2.QM3 is XMITQ.CL2.QM3”.

What to do next

Test the separate transmission queue by sending a message from QM2 to Q1 on QM3 using the queue alias definition Q1A

1. Run the sample program **amqsput** on QM2 to put a message.

```
C:\IBM\MQ>amqsput Q1A QM2
Sample AMQSPUT0 start
target queue is Q1A
Sample request message from QM2 to Q1 using Q1A

Sample AMQSPUT0 end
```

2. Run the sample program **amqsget** to get the message from Q1 on QM3

```
C:\IBM\MQ>amqsget Q1 QM3
Sample AMQSGET0 start
message <Sample request message from QM2 to Q1 using Q1A>
no more messages
Sample AMQSGET0 end
```

Related concepts:

“Access control and multiple cluster transmission queues” on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against SYSTEM.CLUSTER.TRANSMIT.QUEUE, or checking against local profiles for the cluster queue, or cluster queue manager.

“Clustering: Application isolation using multiple cluster transmission queues” on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

“Cluster transmission queues and cluster-sender channels” on page 898
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

Related tasks:

“Adding a queue manager to a cluster: separate transmission queues” on page 926
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

“Creating two-overlapping clusters with a gateway queue manager” on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

“Adding a remote queue definition to isolate messages sent from a gateway queue manager” on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

“Changing the default to separate cluster transmission queues to isolate message traffic” on page 951
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

“Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager” on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

“Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager”
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

“Clustering: Planning how to configure cluster transmission queues” on page 1012
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager

Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

Before you begin

The steps in the task are written to modify the configuration illustrated in Figure 140 on page 932.

1. The gateway queue manager must be on Version 7.5, or later, and on a platform other than z/OS.
2. Construct the overlapping clusters shown in Figure 144 on page 946 in “Creating two-overlapping clusters with a gateway queue manager” on page 946 by following the steps in that task.
3. Do the steps in Figure 140 on page 932in “Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager” on page 931 to create the solution without the additional cluster. Use this as a base for the steps in this task.

About this task

The solution to isolating message traffic to a single application in "Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931 works if the target cluster queue is the only cluster queue on a queue manager. If it is not, you have two choices. Either move the queue to a different queue manager, or create a cluster that isolates the queue from other cluster queues on the queue manager.

This task takes you through the steps to add a cluster to isolate the target queue. The cluster is added just for that purpose. In practice, approach the task of isolating certain applications systematically when you are in the process of designing clusters and cluster naming schemes. Adding a cluster each time a queue requires isolation might end up with many clusters to manage. In this task, you change the configuration in "Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931 by adding a cluster `CL3` to isolate `Q1` on `QM3`. Applications continue to run throughout the change.

The new and changed definitions are highlighted in Figure 141 on page 936. The summary of the changes is as follows: Create a cluster, which means you must also create a new full cluster repository. In the example, `QM3` is made one of the full repositories for `CL3`. Create cluster-sender and cluster-receiver channels for `QM1`to add the gateway queue manager to the new cluster. Change the definition of `Q1` to switch it to `CL3`. Modify the cluster namelist on the gateway queue manager, and add a cluster transmission queue to use the new cluster channel. Finally, switch the queue alias `Q1A` to the new cluster namelist.

IBM WebSphere MQ cannot transfer messages from the transmission queue `XMITQ.CL2.QM3` that you added in "Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931 to the new transmission queue `XMITQ.CL3.QM3`, automatically. It can transfer messages automatically only if both transmission queues are served by the same cluster-sender channel. Instead, the task describes one way to perform the switch manually, which might be appropriate to you. When the transfer is completed, you have the option of reverting to using the default cluster transmission queue for other `CL2` cluster queues on `QM3`. Or you can continue to use `XMITQ.CL2.QM3`. If you decide to revert to a default cluster transmission queue, the gateway queue manager manages the switch for you automatically.

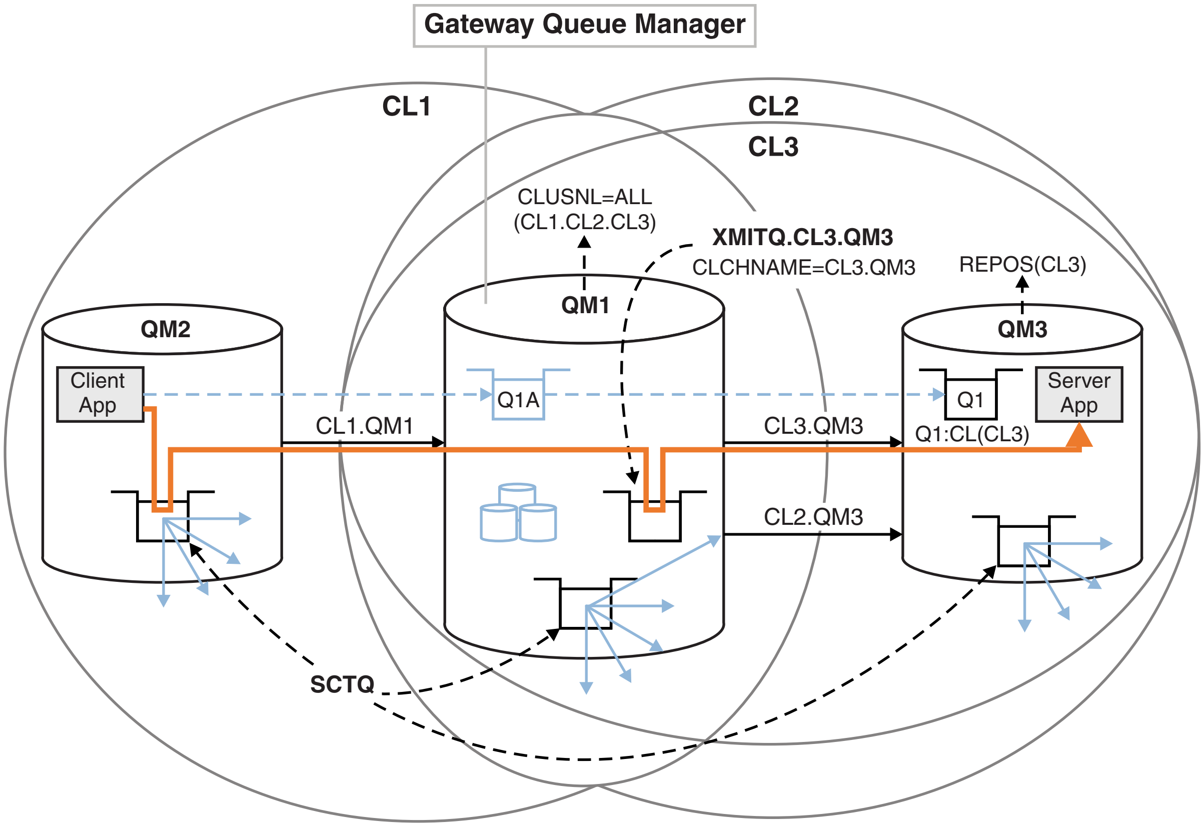


Figure 141. Using an additional cluster to separate message traffic in the gateway queue manager that goes to one of a number of cluster queues on the same queue manager

Procedure

1. Alter the queue managers QM3 and QM5 to make them repositories for both CL2 and CL3.

   To make a queue manager a member of multiple clusters, it must use a cluster name list to identify the clusters it is a member of.

```
*... On QM3 and QM5
DEFINE NAMELIST(CL23) NAMES(CL2, CL3) REPLACE
ALTER QMGR REPOS(' ') REPOSNL(CL23)
```

2. Define the channels between the queue managers QM3 and QM5 for CL3.

```
*... On QM3
DEFINE CHANNEL(CL3.QM5) CHLTYPE(CLUSSDR)  CONNAME('localhost(1415)') CLUSTER(CL3) REPLACE
DEFINE CHANNEL(CL3.QM3) CHLTYPE(CLUSRCVR) CONNAME('localhost(1413)') CLUSTER(CL3) REPLACE

*... On QM5
DEFINE CHANNEL(CL3.QM3) CHLTYPE(CLUSSDR)  CONNAME('localhost(1413)') CLUSTER(CL3) REPLACE
DEFINE CHANNEL(CL3.QM5) CHLTYPE(CLUSRCVR) CONNAME('localhost(1415)') CLUSTER(CL3) REPLACE
```

3. Add the gateway queue manager to CL3.

   Add the gateway queue manager by adding QM1 to CL3 as a partial repository. Create a partial repository by adding cluster-sender and cluster-receiver channels to QM1.

   Also, add CL3 to the name list of all clusters connected to the gateway queue manager.

```
*... On QM1
DEFINE CHANNEL(CL3.QM3) CHLTYPE(CLUSSDR)  CONNAME('localhost(1413)') CLUSTER(CL3) REPLACE
DEFINE CHANNEL(CL3.QM1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1411)') CLUSTER(CL3) REPLACE
ALTER NAMELIST(ALL) NAMES(CL1, CL2, CL3)
```

4. Add a cluster transmission queue to the gateway queue manager, QM1, for messages going to CL3 on QM3.

Initially, stop the cluster-sender channel transferring messages from the transmission queue until you are ready to switch transmission queues.

```
*... On QM1
DEFINE QLOCAL(XMITQ.CL3.QM3) USAGE(XMITQ) CLCHNAME(CL3.QM3) GET(DISABLED) REPLACE
```

5. Drain messages from the existing cluster transmission queue XMITQ.CL2.QM3.

   This subprocedure is intended to preserve the order of messages in Q1 to match the order they arrived at the gateway queue manager. With clusters, message ordering is not fully guaranteed, but is likely. If guaranteed message ordering is required, applications must define the order of messages; see The order in which messages are retrieved from a queue.

   a. Change the target queue Q1 on QM3 from CL2 to CL3.

      ```
      *... On QM3
      ALTER QLOCAL(Q1) CLUSTER(CL3)
      ```

   b. Monitor XMITQ.CL3.QM3 until messages start to be delivered to it.

      Messages start to be delivered to XMITQ.CL3.QM3 when the switch of Q1 to CL3 is propagated to the gateway queue manager.

      ```
      *... On QM1
      DISPLAY QUEUE(XMITQ.CL3.QM3) CURDEPTH
      ```

   c. Monitor XMITQ.CL2.QM3 until it has no messages waiting to be delivered to Q1 on QM3.

      Note: XMITQ.CL2.QM3 might be storing messages for other queues on QM3 that are members of CL2, in which case the depth might not go to zero.

      ```
      *... On QM1
      DISPLAY QUEUE(XMITQ.CL2.QM3) CURDEPTH
      ```

   d. Enable get from the new cluster transmission queue, XMITQ.CL3.QM3

      ```
      *... On QM1
      ALTER QLOCAL(XMITQ.CL3.QM3) GET(ENABLED)
      ```

6. Remove the old cluster transmission queue, XMITQ.CL2.QM3, if it is no longer required.

   Messages for cluster queues in CL2 on QM3 revert to using the default cluster transmission queue on the gateway queue manager, QM1. The default cluster transmission queue is either SYSTEM.CLUSTER.TRANSMIT.QUEUE, or SYSTEM.CLUSTER.TRANSMIT.CL2.QM3. Which one depends on whether the value of the queue manager attribute **DEFCLXQ** on QM1 is SCTQ or CHANNEL. The queue manager transfers messages from XMITQ.CL2.QM3 automatically when the cluster-sender channel CL2.QM3 next starts.

   a. Change the transmission queue, XMITQ.CL2.QM3, from being a cluster transmission queue to being a normal transmission queue.

      This breaks the association of the transmission queue with any cluster-sender channels. In response, IBM WebSphere MQ automatically transfers messages from XMITQ.CL2.QM3 to the default cluster transmission queue when the cluster-sender channel is next started. Until then, messages for CL2 on QM3 continue to be placed on XMITQ.CL2.QM3.

      ```
      *... On QM1
      ALTER QLOCAL(XMITQ.CL2.QM3) CLCHNAME(' ')
      ```

   b. Stop the cluster-sender channel CL2.QM3.

      Stopping and restarting the cluster-sender channel initiates the transfer of messages from XMITQ.CL2.QM3 to the default cluster transmission queue. Typically you would stop and start the channel manually to start the transfer. The transfer starts automatically if the channel restarts after shutting down on the expiry of its disconnect interval.

      ```
      *... On QM1
      STOP CHANNEL(CL2.QM3)
      ```

      The response is that the command is accepted:

      ```
      AMQ8019: Stop WebSphere MQ channel accepted.
      ```

   c. Check that the channel CL2.QM3 is stopped

If the channel does not stop, you can run the **STOP CHANNEL** command again with the FORCE option. An example of setting the FORCE option would be if the channel does not stop, and you cannot restart the other queue manager to synchronize the channel.

```
*... On QM1
DISPLAY CHSTATUS(CL2.QM3)
```

The response is a summary of the channel status

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM3)                       CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1413))               CURRENT
   RQMNAME(QM3)                           STATUS(STOPPED)
   SUBSTATE(MQGET)                        XMITQ(XMITQ.CL2.QM3)
```

d. Start the channel, CL2.QM3.

```
*... On QM1
START CHANNEL(CL2.QM3)
```

The response is that the command is accepted:

```
AMQ8018: Start WebSphere MQ channel accepted.
```

e. Check the channel started.

```
*... On QM1
DISPLAY CHSTATUS(CL2.QM3)
```

The response is a summary of the channel status:

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM3)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1413))   CURRENT
   RQMNAME(QM3)               STATUS(RUNNING)
   SUBSTATE(MQGET)            XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE|CL2.QM3)
```

f. Monitor the gateway queue manager error log for the message “AMQ7341 The transmission queue for channel CL2.QM3 is SYSTEM.CLUSTER.TRANSMIT.*QUEUE|CL2.QM3*”.

g. Delete the cluster transmission queue, XMITQ.CL2.QM3.

```
*... On QM1
DELETE QLOCAL(XMITQ.CL2.QM3)
```

What to do next

Test the separately clustered queue by sending a message from QM2 to Q1 on QM3 using the queue alias definition Q1A

1. Run the sample program **amqsput** on QM2 to put a message.

```
C:\IBM\MQ>amqsput Q1A QM2
Sample AMQSPUT0 start
target queue is Q1A
Sample request message from QM2 to Q1 using Q1A

Sample AMQSPUT0 end
```

2. Run the sample program **amqsget** to get the message from Q1 on QM3

```
C:\IBM\MQ>amqsget Q1 QM3
Sample AMQSGET0 start
message <Sample request message from QM2 to Q1 using Q1A>
no more messages
Sample AMQSGET0 end
```

Related concepts:

"Access control and multiple cluster transmission queues" on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, or checking against local profiles for the cluster queue, or cluster queue manager.

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

"Cluster transmission queues and cluster-sender channels" on page 898
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

Related tasks:

"Adding a queue manager to a cluster: separate transmission queues" on page 926
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

"Changing the default to separate cluster transmission queues to isolate message traffic" on page 951
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

"Clustering: Planning how to configure cluster transmission queues" on page 1012
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Adding a queue manager to a cluster by using DHCP

Add a queue manager to a cluster, using DHCP. The task demonstrates omitting `CONNAME` value on a `CLUSRCVR` definition.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

The task demonstrates two special features:
- The ability to omit the CONNAME value on a CLUSRCVR definition.
- The ability to use +QMNAME+ on a CLUSSDR definition.

Neither feature is provided on z/OS.

Scenario:
- The INVENTORY cluster has been set up as described in "Setting up a new cluster" on page 913. It contains two queue managers, LONDON and NEWYORK, which both hold full repositories.
- A new branch of the chain store is being set up in Paris and you want to add a queue manager called PARIS to the cluster.
- Queue manager PARIS sends inventory updates to the application running on the system in New York by putting messages on the INVENTQ queue.
- Network connectivity exists between all three systems.
- The network protocol is TCP.
- The PARIS queue manager system uses DHCP, which means that the IP addresses might change on system restart.
- The channels between the PARIS and LONDON systems are named according to a defined naming convention. The convention uses the queue manager name of the full repository queue manager on LONDON.
- Administrators of the PARIS queue manager have no information about the name of the queue manager on the LONDON repository. The name of the queue manager on the LONDON repository is subject to change.

About this task

Follow these steps to add a queue manager to a cluster by using DHCP.

Procedure

1. Decide which full repository PARIS refers to first.

   Every queue manager in a cluster must refer to one or other of the full repositories. It gathers information about the cluster from a full repository and so builds up its own partial repository. Choose either of the repositories as the full repository. As soon as a new queue manager is added to the cluster it immediately learns about the other repository as well. Information about changes to a queue manager is sent directly to two repositories. In this example we choose to link PARIS to the queue manager LONDON, purely for geographical reasons.

   Note: Perform the remaining steps in any order, after queue manager PARIS is started.

2. Define a CLUSRCVR channel on queue manager PARIS.

   Every queue manager in a cluster needs to define a cluster-receiver channel on which it can receive messages. On PARIS, define:

   ```
   DEFINE CHANNEL(INVENTORY.PARIS) CHLTYPE(CLUSRCVR)
   TRPTYPE(TCP) CLUSTER(INVENTORY)
   DESCR('Cluster-receiver channel for queue manager PARIS')
   ```

   The cluster-receiver channel advertises the availability of the queue manager to receive messages from other queue managers in the cluster INVENTORY. You do not need to specify the CONNAME on the cluster-receiver channel. You can request IBM WebSphere MQ to find out the connection name from the system, either by omitting CONNAME, or by specifying CONNAME(' '). IBM WebSphere MQ generates the CONNAME value using the current IP address of the system; see CONNAME. There is no need to make definitions on other queue managers for a sending end to the cluster-receiver channel INVENTORY.PARIS. Other definitions are made automatically when needed.

3. Define a CLUSSDR channel on queue manager PARIS.

   Every queue manager in a cluster needs to define one cluster-sender channel on which it can send messages to its initial full repository. On PARIS, make the following definition for a channel called INVENTORY.+QMNAME+ to the queue manager with the network address LONDON.CHSTORE.COM.

   ```
   DEFINE CHANNEL(INVENTORY.+QMNAME+) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(LONDON.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-sender channel from PARIS to repository at LONDON')
   ```

4. Optional: If this queue manager is rejoining a cluster, complete some extra steps.

   a. If you are adding a queue manager to a cluster that has previously been removed from the same cluster, check that it is now showing as a cluster member. If not, complete the following extra steps:

      1) Issue the **REFRESH CLUSTER** command on the queue manager you are adding. This step stops the cluster channels, and gives your local cluster cache a fresh set of sequence numbers that are assured to be up-to-date within the rest of the cluster.

         ```
         REFRESH CLUSTER(INVENTORY) REPOS(YES)
         ```

         Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

      2) Restart the CLUSSDR channel (for example, using the START CHANNEL command).

      3) Restart the CLUSRCVR channel.

   b. If the cluster is a publish/subscribe cluster, and the rejoining queue manager has subscriptions, issue the following command to ensure the proxy subscriptions are correctly synchronized across the cluster:

      ```
      REFRESH QMGR TYPE(PROXYSUB)
      ```

Results

The cluster set up by this task is the same as for “Adding a queue manager to a cluster” on page 924:

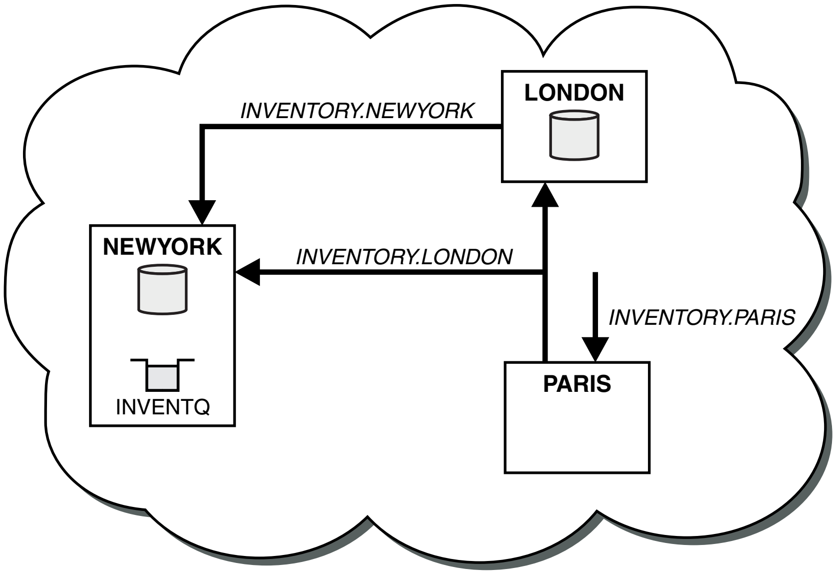


Figure 142. The INVENTORY cluster with three queue managers

By making only two definitions, a CLUSRCVR definition and a CLUSSDR definition, we have added the queue manager PARIS to the cluster.

On the PARIS queue manager, the CLUSSDR containing the string +QMNAME+ starts. On the LONDON system IBM WebSphere MQ resolves the +QMNAME+ to the queue manager name (LONDON). IBM WebSphere MQ then matches the definition for a channel called INVENTORY.LONDON to the corresponding CLUSRCVR definition.

WebSphere MQ sends back the resolved channel name to the PARIS queue manager. At PARIS, the CLUSSDR channel definition for the channel called INVENTORY.+QMNAME+ is replaced by an internally generated CLUSSDR definition for INVENTORY.LONDON. This definition contains the resolved channel name, but otherwise is the same as the +QMNAME+ definition that you made. The cluster repositories are also brought up to date with the channel definition with the newly resolved channel name.

Note:

1. The channel created with the +QMNAME+ name becomes inactive immediately. It is never used to transmit data.
2. Channel exits might see the channel name change between one invocation and the next.

Now the PARIS queue manager learns, from the repository at LONDON, that the INVENTQ queue is hosted by queue manager NEWYORK. When an application hosted by the system in Paris tries to put messages to the INVENTQ, PARIS automatically defines a cluster-sender channel to connect to the cluster-receiver channel INVENTORY.NEWYORK. The application can receive responses when its queue-manager name is specified as the target queue manager and a reply-to queue is provided.

Related information:

DEFINE CHANNEL

Adding a queue manager that hosts a queue

Add another queue manager to the cluster, to host another INVENTQ queue. Requests are sent alternately to the queues on each queue manager. No changes need to be made to the existing INVENTQ host.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster has been set up as described in "Adding a queue manager to a cluster" on page 924. It contains three queue managers; LONDON and NEWYORK both hold full repositories, PARIS holds a partial repository. The inventory application runs on the system in New York, connected to the NEWYORK queue manager. The application is driven by the arrival of messages on the INVENTQ queue.
- A new store is being set up in Toronto. To provide additional capacity you want to run the inventory application on the system in Toronto as well as New York.
- Network connectivity exists between all four systems.
- The network protocol is TCP.

Note: The queue manager TORONTO contains only a partial repository. If you want to add a full-repository queue manager to a cluster, refer to "Moving a full repository to another queue manager" on page 958.

About this task

Follow these steps to add a queue manager that hosts a queue.

Procedure

1. Decide which full repository TORONTO refers to first.

   Every queue manager in a cluster must refer to one or other of the full repositories. It gathers information about the cluster from a full repository and so builds up its own partial repository. It is of no particular significance which repository you choose. In this example, we choose NEWYORK. Once the new queue manager has joined the cluster it communicates with both of the repositories.
2. Define the CLUSRCVR channel.

   Every queue manager in a cluster needs to define a cluster-receiver channel on which it can receive messages. On TORONTO, define a CLUSRCVR channel:

   ```
   DEFINE CHANNEL(INVENTORY.TORONTO) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(TORONTO.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-receiver channel for TORONTO')
   ```

   The TORONTO queue manager advertises its availability to receive messages from other queue managers in the INVENTORY cluster using its cluster-receiver channel.
3. Define a CLUSSDR channel on queue manager TORONTO.

   Every queue manager in a cluster needs to define one cluster-sender channel on which it can send messages to its first full repository. In this case choose NEWYORK. TORONTO needs the following definition:

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-sender channel from TORONTO to repository at NEWYORK')
   ```
4. Optional: If this queue manager is rejoining a cluster, complete some extra steps.

a. If you are adding a queue manager to a cluster that has previously been removed from the same cluster, check that it is now showing as a cluster member. If not, complete the following extra steps:

   1) Issue the **REFRESH CLUSTER** command on the queue manager you are adding. This step stops the cluster channels, and gives your local cluster cache a fresh set of sequence numbers that are assured to be up-to-date within the rest of the cluster.

      ```
      REFRESH CLUSTER(INVENTORY) REPOS(YES)
      ```

      Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

   2) Restart the CLUSSDR channel (for example, using the START CHANNEL command).

   3) Restart the CLUSRCVR channel.

b. If the cluster is a publish/subscribe cluster, and the rejoining queue manager has subscriptions, issue the following command to ensure the proxy subscriptions are correctly synchronized across the cluster:

   ```
   REFRESH QMGR TYPE(PROXYSUB)
   ```

5. Review the inventory application for message affinities.

   Before proceeding, ensure that the inventory application does not have any dependencies on the sequence of processing of messages and install the application on the system in Toronto.

6. Define the cluster queue INVENTQ.

   The INVENTQ queue, which is already hosted by the NEWYORK queue manager, is also to be hosted by TORONTO. Define it on the TORONTO queue manager as follows:

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
   ```

Results

Figure 143 on page 945 shows the INVENTORY cluster set up by this task.

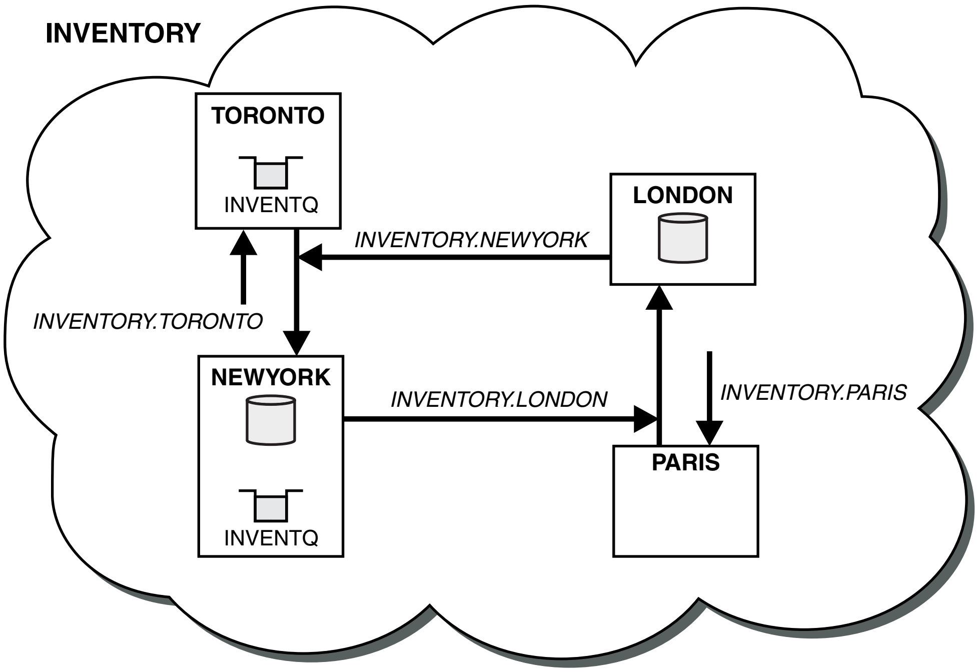


Figure 143. The `INVENTORY` cluster with four queue managers

The `INVENTQ` queue and the inventory application are now hosted on two queue managers in the cluster. This increases their availability, speeds up throughput of messages, and allows the workload to be distributed between the two queue managers. Messages put to `INVENTQ` by either `TORONTO` or `NEWYORK` are handled by the instance on the local queue manager whenever possible. Messages put by `LONDON` or `PARIS` are routed alternately to `TORONTO` or `NEWYORK`, so that the workload is balanced.

This modification to the cluster was accomplished without you having to alter the definitions on queue managers `NEWYORK`, `LONDON`, and `PARIS`. The full repositories in these queue managers are updated automatically with the information they need to be able to send messages to `INVENTQ` at `TORONTO`. The inventory application continues to function if one of the `NEWYORK` or the `TORONTO` queue manager becomes unavailable, and it has sufficient capacity. The inventory application must be able to work correctly if it is hosted in both locations.

As you can see from the result of this task, you can have the same application running on more than one queue manager. You can clustering to distribution workload evenly.

An application might not be able to process records in both locations. For example, suppose that you decide to add a customer-account query and update application running in `LONDON` and `NEWYORK`. An account record can only be held in one place. You could decide to control the distribution of requests by using a data partitioning technique. You can split the distribution of the records. You could arrange for half the records, for example for account numbers 00000 - 49999, to be held in `LONDON`. The other half, in the range 50000 - 99999, are held in `NEWYORK`. You could then write a cluster workload exit program to examine the account field in all messages, and route the messages to the appropriate queue manager.

What to do next

Now that you have completed all the definitions, if you have not already done so start the channel initiator on IBM WebSphere MQ for z/OS. On all platforms, start a listener program on queue manager `TORONTO`. The listener program waits for incoming network requests and starts the cluster-receiver channel when it is needed.

Creating two-overlapping clusters with a gateway queue manager

Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

About this task

The example cluster configuration used to illustrate isolating cluster message traffic is shown in Figure 144. The example is described in “Clustering: Application isolation using multiple cluster transmission queues” on page 1009.

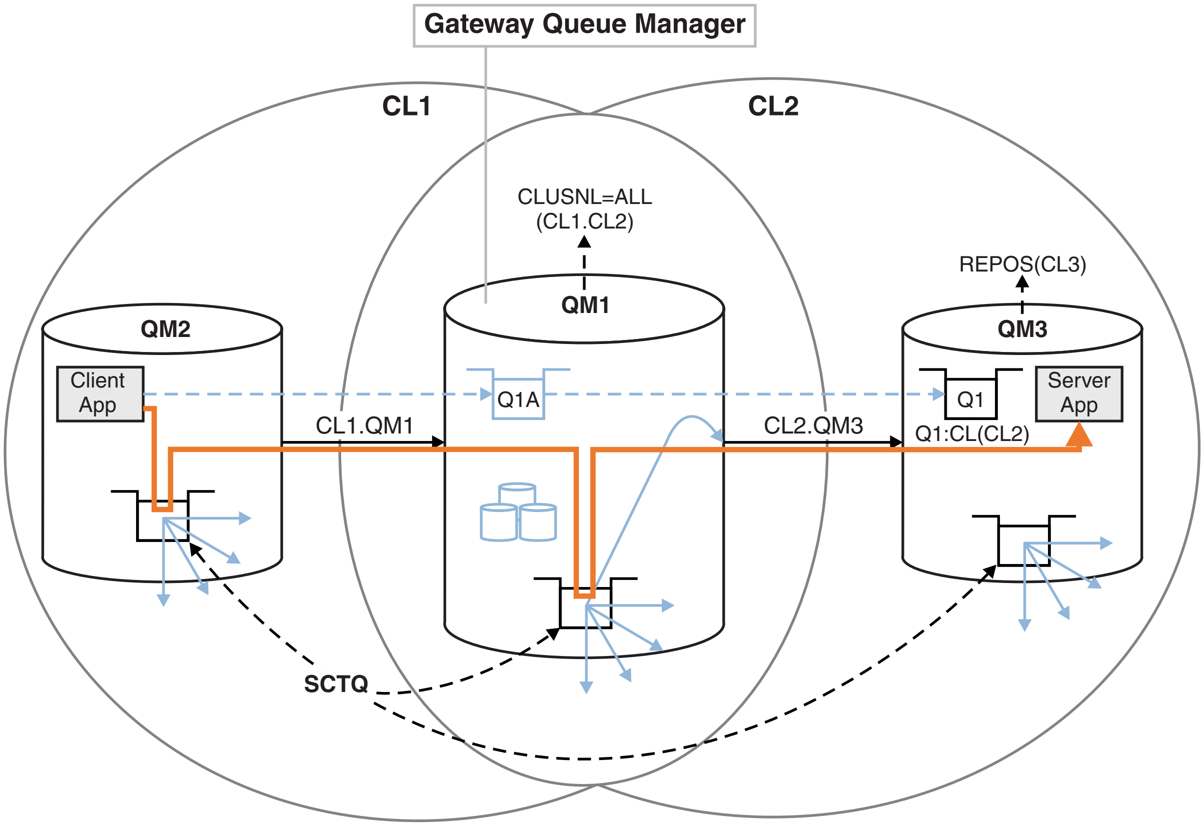


Figure 144. Client-server application deployed to hub and spoke architecture using IBM WebSphere MQ clusters

To make the number of steps to construct the example as few as possible, the configuration is kept simple, rather than realistic. The example might represent the integration of two clusters created by two separate organizations. For a more realistic scenario, see “Clustering: Planning how to configure cluster transmission queues” on page 1012.

Follow the steps to construct the clusters. The clusters are used in the following examples of isolating the message traffic from the client application to the server application.

The instructions add a couple of extra queue managers so that each cluster has two repositories. The gateway queue manager is not used as a repository for performance reasons.

Procedure

1. Create and start the queue managers QM1, QM2, QM3, QM4, QM5.

```
crtmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE QMn
strmqm QmgrName
```

Note: QM4 and QM5 are the backup full repositories for the clusters.

2. Define and start listeners for each of the queue managers.

   ```
   *... On QMn
   DEFINE LISTENER(TCP141n) TRPTYPE(TCP) IPADDR(hostname) PORT(141n) CONTROL(QMGR) REPLACE
   START LISTENER(TCP141n)
   ```

3. Create a cluster name list for all of the clusters.

   ```
   *... On QM1
   DEFINE NAMELIST(ALL) NAMES(CL1, CL2) REPLACE
   ```

4. Make QM2 and QM4 full repositories for CL1, QM3 and QM5 full repositories for CL2.

   a. For CL1:

   ```
   *... On QM2 and QM4
   ALTER QMGR REPOS(CL1) DEFCLXQ(SCTQ)
   ```

   b. For CL2:

   ```
   *... On QM3 and QM5
   ALTER QMGR REPOS(CL2) DEFCLXQ(SCTQ)
   ```

5. Add the cluster-sender and cluster-receiver channels for each queue manager and cluster.

   Run the following commands on QM2, QM3, QM4 and QM5, where *c*, *n*, and *m* take the values shown in Table 127 for each queue manager:

Table 127. Parameter values for creating clusters 1 and 2

Queue manager	Cluster *c*	Other repository *n*	This repository *m*
QM2	1	4	2
QM4	1	2	4
QM3	2	5	3
QM5	2	3	5

   ```
   *... On QMm
   DEFINE CHANNEL(CLc.QMn) CHLTYPE(CLUSSDR)  CONNAME('localhost(141n)') CLUSTER(CLc) REPLACE
   DEFINE CHANNEL(CLc.QMm) CHLTYPE(CLUSRCVR) CONNAME('localhost(141m)') CLUSTER(CLc) REPLACE
   ```

6. Add the gateway queue manager, QM1, to each of the clusters.

   ```
   *... On QM1
   DEFINE CHANNEL(CL1.QM2) CHLTYPE(CLUSSDR)  CONNAME('localhost(1412)') CLUSTER(CL1) REPLACE
   DEFINE CHANNEL(CL1.QM1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1411)') CLUSTER(CL1) REPLACE
   DEFINE CHANNEL(CL2.QM3) CHLTYPE(CLUSSDR)  CONNAME('localhost(1413)') CLUSTER(CL2) REPLACE
   DEFINE CHANNEL(CL2.QM1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1411)') CLUSTER(CL2) REPLACE
   ```

7. Add the local queue Q1 to queue manager QM3 in cluster CL2.

   ```
   *... On QM3
   DEFINE QLOCAL(Q1) CLUSTER(CL2) REPLACE
   ```

8. Add the clustered queue manager alias Q1A to the gateway queue manager.

   ```
   *... On QM1
   DEFINE QALIAS(Q1A)  CLUSNL(ALL) TARGET(Q1) TARGTYPE(QUEUE) DEFBIND(NOTFIXED) REPLACE
   ```

   Note: Applications using the queue manager alias on any other queue manager but QM1, must specify DEFBIND(NOTFIXED) when they open the alias queue. **DEFBIND** specifies whether the routing information in the message header is fixed when the queue is opened by the application. If it is set to the default value, OPEN, messages are routed to Q1@QM1. Q1@QM1 does not exist, so messages from other queue managers end up on a dead letter queue. By setting the queue attribute to DEFBIND(NOTFIXED), applications such as **amqsput**, which default to the queue setting of **DEFBIND**, behave in the correct way.

9. Add the cluster queue manager alias definitions for all the clustered queue managers to the gateway queue manager, QM1.

```
*... On QM1
DEFINE QREMOTE(QM2) RNAME(' ') RQMNAME(QM2) CLUSNL(ALL) REPLACE
DEFINE QREMOTE(QM3) RNAME(' ') RQMNAME(QM3) CLUSNL(ALL) REPLACE
```

Tip: The queue manager alias definitions on the gateway queue manager transfer messages that refer to a queue manager in another cluster; see Clustered queue manager aliases.

What to do next

1. Test the queue alias definition by sending a message from QM2 to Q1 on QM3 using the queue alias definition Q1A.
   a. Run the sample program **amqsput** on QM2 to put a message.

   ```
   C:\IBM\MQ>amqsput Q1A QM2
   Sample AMQSPUT0 start
   target queue is Q1A
   Sample request message from QM2 to Q1 using Q1A

   Sample AMQSPUT0 end
   ```

   b. Run the sample program **amqsget** to get the message from Q1 on QM3

   ```
   C:\IBM\MQ>amqsget Q1 QM3
   Sample AMQSGET0 start
   message <Sample request message from QM2 to Q1 using Q1A>
   no more messages
   Sample AMQSGET0 end
   ```

2. Test the queue manager alias definitions by sending a request message and receiving a reply message on a temporary-dynamic reply queue.

   The diagram shows the path taken by the reply message back to a temporary dynamic queue, which is called RQ. The server application, connected to QM3, opens the reply queue using the queue manager name QM2. The queue manager name QM2 is defined as a clustered queue manager alias on QM1. QM3 routes the reply message to QM1. QM1 routes the message to QM2.

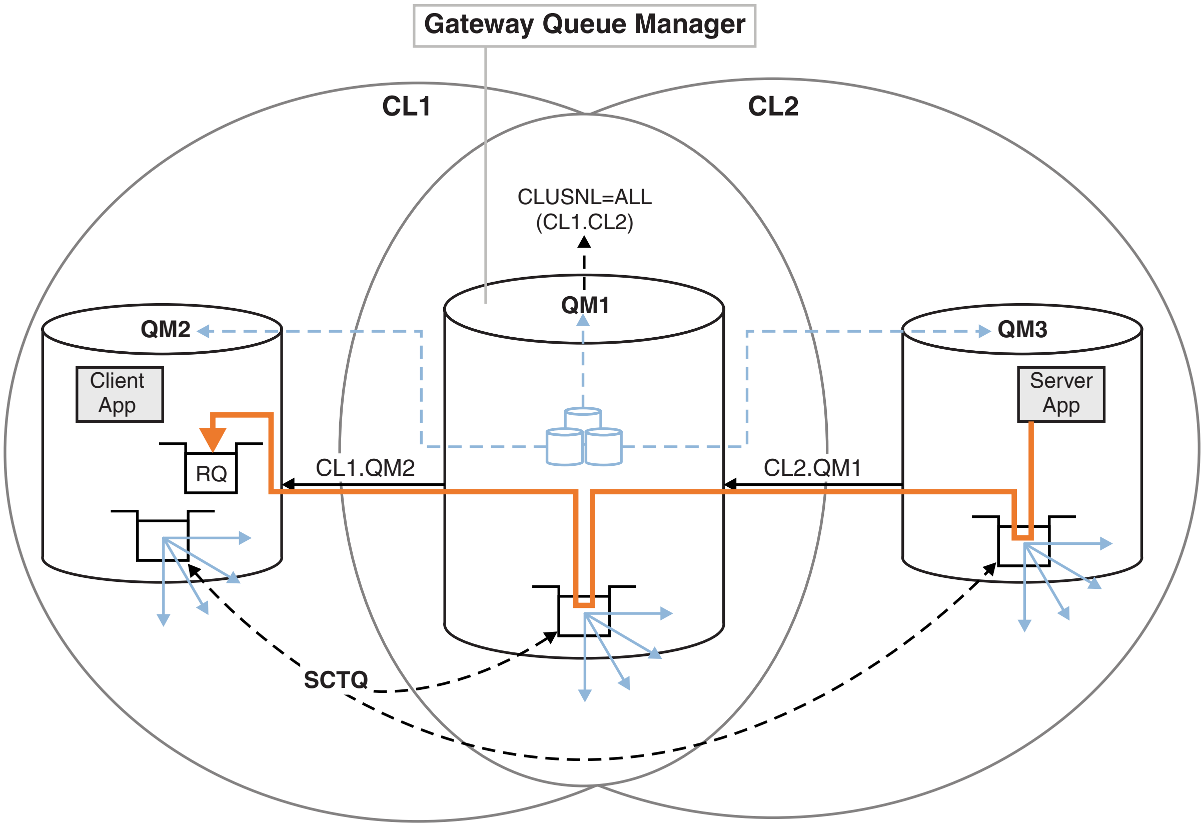


Figure 145. Using a queue manager alias to return the reply message to a different cluster

The way the routing works is as follows. Every queue manager in each cluster has a queue manager alias definition on QM1. The aliases are clustered in all the clusters. The grey dashed arrows from each of the aliases to a queue manager show that each queue manager alias is resolved to a real queue manager in at least one of the clusters. In this case, the QM2 alias is clustered in both cluster CL1 and CL2, and is resolved to the real queue manager QM2 in CL1. The server application creates the reply message using the reply to queue name RQ, and reply to queue manager name QM2. The message is routed to QM1 because the queue manager alias definition QM2 is defined on QM1 in cluster CL2 and queue manager QM2 is not in cluster CL2. As the message cannot be sent to the target queue manager, it is sent to the queue manager that has the alias definition.

QM1 places the message on the cluster transmission queue on QM1 for transferal to QM2. QM1 routes the message to QM2 because the queue manager alias definition on QM1 for QM2 defines QM2 as the real target queue manager. The definition is not circular, because alias definitions can refer only to real definitions; the alias cannot point to itself. The real definition is resolved by QM1, because both QM1 and QM2 are in the same cluster, CL1. QM1 finds out the connection information for QM2 from the repository for CL1, and routes the message to QM2. For the message to be rerouted by QM1, the server application must have opened the reply queue with the option DEFBIND set to MQBND_BIND_NOT_FIXED. If the server application had opened the reply queue with the option MQBND_BIND_ON_OPEN, the message is not rerouted and ends up on a dead letter queue.

a. Create a clustered request queue with a trigger on QM3.

```
*... On QM3
DEFINE QLOCAL(QR) CLUSTER(CL2) TRIGGER INITQ(SYSTEM.DEFAULT.INITIATION.QUEUE) PROCESS(ECHO) REPLACE
```

b. Create a clustered queue alias definition of QR on the gateway queue manager, QM1.

```
*... On QM1
DEFINE QALIAS(QRA) CLUSNL(ALL) TARGET(QR) TARGTYPE(QUEUE) DEFBIND(NOTFIXED) REPLACE
```

c. Create a process definition to start the sample echo program **amqsech** on QM3.

```
*... On QM3
DEFINE PROCESS(ECHO) APPLICID(AMQSECH) REPLACE
```

d. Create a model queue on QM2 for the sample program **amqsreq** to create the temporary-dynamic reply queue.

```
*... On QM2
DEFINE QMODEL(SYSTEM.SAMPLE.REPLY) REPLACE
```

e. Test the queue manager alias definition by sending a request from QM2 to QR on QM3 using the queue alias definition QRA.

1) Run the trigger monitor program on QM3.

```
runmqtrm -m QM3
```

The output is

```
C:\IBM\MQ>runmqtrm -m QM3
5724-H72 (C) Copyright IBM Corp. 1994, 2011. ALL RIGHTS RESERVED.
01/02/2012  16:17:15: WebSphere MQ trigger monitor started.

__________________________________________________
01/02/2012  16:17:15: Waiting for a trigger message
```

2) Run the sample program **amqsreq** on QM2 to put a request and wait for a reply.

```
C:\IBM\MQ>amqsreq QRA QM2
Sample AMQSREQ0 start
server queue is QRA
replies to 4F2961C802290020
A request message from QM2 to QR on QM3

response <A request message from QM2 to QR on QM3>
no more replies
Sample AMQSREQ0 end
```

Related concepts:

"Access control and multiple cluster transmission queues" on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against SYSTEM.CLUSTER.TRANSMIT.QUEUE, or checking against local profiles for the cluster queue, or cluster queue manager.

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

Related tasks:

"Adding a queue manager to a cluster: separate transmission queues" on page 926
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue

remote definition, and a separate sender channel and transmission queue.

"Changing the default to separate cluster transmission queues to isolate message traffic"
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

"Clustering: Planning how to configure cluster transmission queues" on page 1012
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Changing the default to separate cluster transmission queues to isolate message traffic

You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

Before you begin

1. The gateway queue manager must be on Version 7.5, or later, and on a platform other than z/OS.
2. Construct the overlapping clusters shown in Figure 144 on page 946 in "Creating two-overlapping clusters with a gateway queue manager" on page 946 by following the steps in that task.

About this task

To implement the architecture with multiple clusters queue, your gateway queue manager must be on Version 7.5, or later. All you do to use multiple cluster transmission queues is to change the default cluster transmission queue type on the gateway queue manager. Change the value of the queue manager attribute **DEFCLXQ** on QM1 from SCTQ to CHANNEL; see Figure 146 on page 952. The diagram shows one message flow. For flows to other queue managers, or to other clusters, the queue manager creates additional permanent dynamic cluster transmission queues. Each cluster-sender channel transfers messages from a different cluster transmission queue.

The change does not take effect immediately, unless you are connecting the gateway queue manager to clusters for the first time. The task includes steps for the typical case of managing a change to an existing configuration. To set up a queue manager to use separate cluster transmission queues when it first joins a cluster; see "Adding a queue manager to a cluster: separate transmission queues" on page 926.

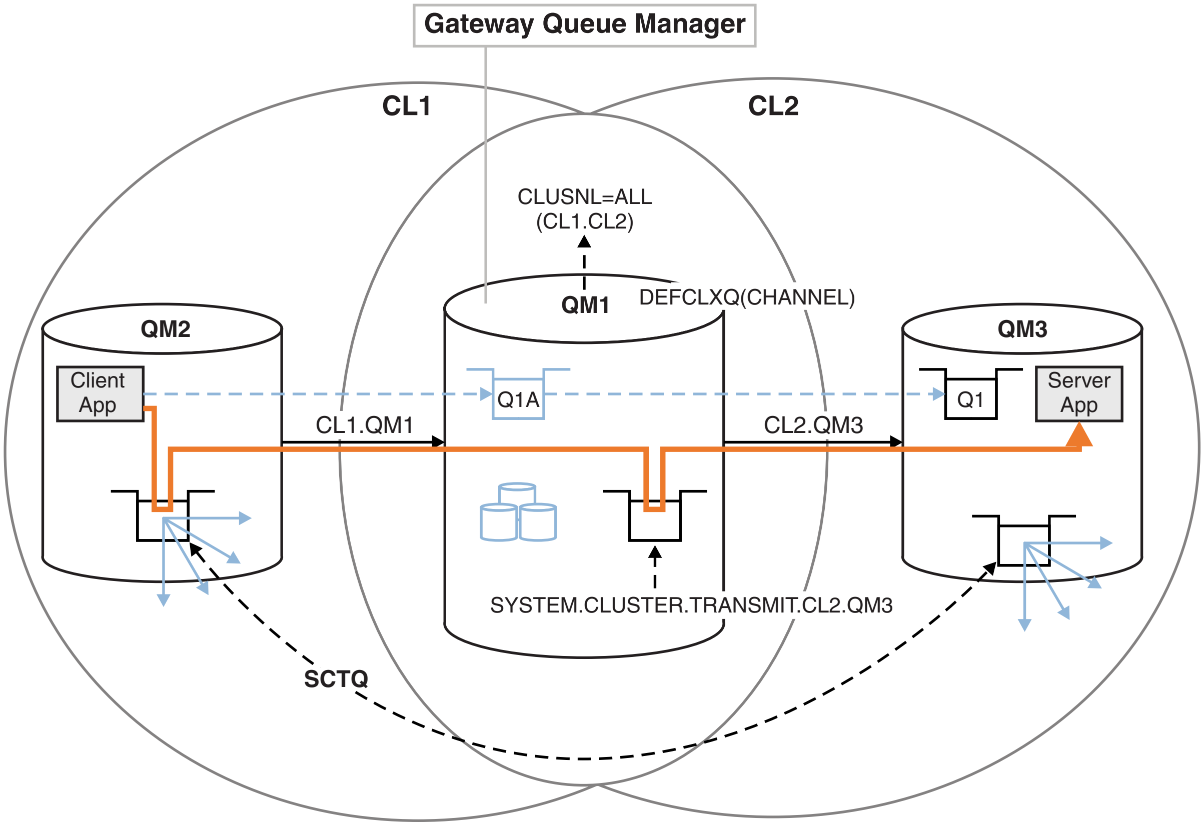


Figure 146. Client-server application deployed to hub and spoke architecture with separate cluster transmission queues on the gateway queue manager.

Procedure

1. Change the gateway queue manager to use separate cluster transmission queues.

   ```
   *... On QM1
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

2. Switch to the separate cluster transmission queues.

   Any cluster-sender channel that is not running switches to using separate cluster transmission queues when it next starts.

   To switch the running channels, either restart the queue manager, or follow these steps:

   a. List the cluster-sender channels that are running with SYSTEM.CLUSTER.TRANSMIT.QUEUE.

      ```
      *... On QM1
      DISPLAY CHSTATUS(*) WHERE(XMITQ EQ 'SYSTEM.CLUSTER.TRANSMIT.QUEUE')
      ```

      The response is list of channel status reports:

      ```
      AMQ8417: Display Channel Status details.
         CHANNEL(CL1.QM2)              CHLTYPE(CLUSSDR)
         CONNAME(127.0.0.1(1412))      CURRENT
         RQMNAME(QM2)                  STATUS(RUNNING)
         SUBSTATE(MQGET)               XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
      AMQ8417: Display Channel Status details.
         CHANNEL(CL2.QM3)              CHLTYPE(CLUSSDR)
         CONNAME(127.0.0.1(1413))      CURRENT
         RQMNAME(QM3)                  STATUS(RUNNING)
         SUBSTATE(MQGET)               XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
      ```

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM5)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1415))   CURRENT
   RQMNAME(QM5)               STATUS(RUNNING)
   SUBSTATE(MQGET)            XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
AMQ8417: Display Channel Status details.
   CHANNEL(CL1.QM4)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1414))   CURRENT
   RQMNAME(QM4)               STATUS(RUNNING)
   SUBSTATE(MQGET)            XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
```

b. Stop the channels that are running

For each channel in the list, run the command:

```
*... On QM1
STOP CHANNEL(ChannelName)
```

Where *ChannelName* is each of CL1.QM2, CL1.QM4, CL1.QM3, CL1.QM5.

The response is that the command is accepted:

```
AMQ8019: Stop WebSphere MQ channel accepted.
```

c. Monitor which channels are stopped

```
*... On QM1
DISPLAY CHSTATUS(*) WHERE(XMITQ EQ 'SYSTEM.CLUSTER.TRANSMIT.QUEUE')
```

The response is a list of channels that are still running and channels that are stopped:

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL1.QM2)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1412))   CURRENT
   RQMNAME(QM2)               STATUS(STOPPED)
   SUBSTATE( )                XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM3)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1413))   CURRENT
   RQMNAME(QM3)               STATUS(STOPPED)
   SUBSTATE( )                XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM5)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1415))   CURRENT
   RQMNAME(QM5)               STATUS(STOPPED)
   SUBSTATE( )                XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
AMQ8417: Display Channel Status details.
   CHANNEL(CL1.QM4)           CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1414))   CURRENT
   RQMNAME(QM4)               STATUS(STOPPED)
   SUBSTATE( )                XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
```

d. Start each stopped channel.

Do this step for all the channels that were running. If a channel does not stop, you can run the **STOP CHANNEL** command again with the FORCE option. An example of setting the FORCE option would be if the channel does not stop, and you cannot restart the other queue manager to synchronize the channel.

```
*... On QM1
START CHANNEL(CL2.QM5)
```

The response is that the command is accepted:

```
AMQ8018: Start WebSphere MQ channel accepted.
```

e. Monitor the transmission queues being switched.

   Monitor the gateway queue manager error log for the message “AMQ7341 The transmission queue for channel CL2.QM3 is SYSTEM.CLUSTER.TRANSMIT.*QUEUE|CL2.QM3*”.

f. Check that SYSTEM.CLUSTER.TRANSMIT.QUEUE is no longer used

```
*... On QM1
DISPLAY CHSTATUS(*) WHERE(XMITQ EQ 'SYSTEM.CLUSTER.TRANSMIT.QUEUE')
DISPLAY QUEUE(SYSTEM.CLUSTER.TRANSMIT.QUEUE) CURDEPTH
```

   The response is list of channel status reports, and the depth of SYSTEM.CLUSTER.TRANSMIT.QUEUE:

```
AMQ8420: Channel Status not found.
AMQ8409: Display Queue details.
   QUEUE(SYSTEM.CLUSTER.TRANSMIT.QUEUE)    TYPE(QLOCAL)
   CURDEPTH(0)
```

g. Monitor which channels are started

```
*... On QM1
DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
```

   The response is a list of the channels, in this case already running with the new default cluster transmission queues:

```
AMQ8417: Display Channel Status details.
   CHANNEL(CL1.QM2)                        CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1412))                CURRENT
   RQMNAME(QM2)                            STATUS(RUNNING)
   SUBSTATE(MQGET)
   XMITQ(SYSTEM.CLUSTER.TRANSMIT.CL1.QM2)
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM3)                        CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1413))                CURRENT
   RQMNAME(QM3)                            STATUS(RUNNING)
   SUBSTATE(MQGET)
   XMITQ(SYSTEM.CLUSTER.TRANSMIT.CL2.QM3)
AMQ8417: Display Channel Status details.
   CHANNEL(CL2.QM5)                        CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1415))                CURRENT
   RQMNAME(QM5)                            STATUS(RUNNING)
   SUBSTATE(MQGET)
   XMITQ(SYSTEM.CLUSTER.TRANSMIT.CL2.QM5)
AMQ8417: Display Channel Status details.
   CHANNEL(CL1.QM4)                        CHLTYPE(CLUSSDR)
   CONNAME(127.0.0.1(1414))                CURRENT
   RQMNAME(QM4)                            STATUS(RUNNING)
   SUBSTATE(MQGET)
   XMITQ(SYSTEM.CLUSTER.TRANSMIT.CL1.QM4)
```

What to do next

1. Test the automatically defined cluster transmission queue by sending a message from QM2 to Q1 on QM3, resolving queue name with the queue alias definition Q1A

   a. Run the sample program **amqsput** on QM2 to put a message.

```
C:\IBM\MQ>amqsput Q1A QM2
Sample AMQSPUT0 start
target queue is Q1A
```

```
Sample request message from QM2 to Q1 using Q1A

Sample AMQSPUT0 end
```

   b. Run the sample program **amqsget** to get the message from Q1 on QM3

```
C:\IBM\MQ>amqsget Q1 QM3
Sample AMQSGET0 start
message <Sample request message from QM2 to Q1 using Q1A>
no more messages
Sample AMQSGET0 end
```

2. Consider whether to reconfigure security, by configuring security for the cluster queues on the queue managers where messages for the cluster queues originate.

Related concepts:

"Access control and multiple cluster transmission queues" on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against SYSTEM.CLUSTER.TRANSMIT.QUEUE, or checking against local profiles for the cluster queue, or cluster queue manager.

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

Related tasks:

"Adding a queue manager to a cluster: separate transmission queues" on page 926
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using multiple cluster transmission queues.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

"Changing the default to separate cluster transmission queues to isolate message traffic" on page 951
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

"Clustering: Planning how to configure cluster transmission queues" on page 1012
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Removing a cluster queue from a queue manager

Disable the INVENTQ queue at Toronto. Send all the inventory messages to New York, and delete the INVENTQ queue at Toronto when it is empty.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster has been set up as described in "Adding a queue manager that hosts a queue" on page 943. It contains four queue managers. LONDON and NEWYORK both hold full repositories. PARIS and TORONTO hold partial repositories. The inventory application runs on the systems in New York and Toronto and is driven by the arrival of messages on the INVENTQ queue.
- Because of reduced workload, you no longer want to run the inventory application in Toronto. You want to disable the INVENTQ queue hosted by the queue manager TORONTO, and have TORONTO feed messages to the INVENTQ queue in NEWYORK.
- Network connectivity exists between all four systems.
- The network protocol is TCP.

About this task

Follow these steps to remove a cluster queue.

Procedure

1. Indicate that the queue is no longer available.

   To remove a queue from a cluster, remove the cluster name from the local queue definition. Alter the INVENTQ on TORONTO so that it is not accessible from the rest of the cluster:

   ```
   ALTER QLOCAL(INVENTQ) CLUSTER(' ')
   ```
2. Check that the queue is no longer available.

   On a full repository queue manager, either LONDON or NEWYORK, check that the queue is no longer hosted by queue manager TORONTO by issuing the following command:

   ```
   DIS QCLUSTER (INVENTQ)
   ```

   TORONTO is not listed in the results, if the ALTER command has completed successfully.
3. Disable the queue.

   Disable the INVENTQ queue at TORONTO so that no further messages can be written to it:

   ```
   ALTER QLOCAL(INVENTQ) PUT(DISABLED)
   ```

   Now messages in transit to this queue using MQOO_BIND_ON_OPEN go to the dead-letter queue. You need to stop all applications from putting messages explicitly to the queue on this queue manager.
4. Monitor the queue until it is empty.

   Monitor the queue using the DISPLAY QUEUE command, specifying the attributes IPPROCS, OPPROCS, and CURDEPTH, or use the **WRKMQMSTS** command on IBM i. When the number of input and output processes, and the current depth of the queuesare all zero, the queue is empty.
5. Monitor the channel to ensure there are no in-doubt messages.

To be sure that there are no in-doubt messages on the channel INVENTORY.TORONTO, monitor the cluster-sender channel called INVENTORY.TORONTO on each of the other queue managers. Issue the DISPLAY CHSTATUS command specifying the INDOUBT parameter from each queue manager:

```
DISPLAY CHSTATUS(INVENTORY.TORONTO) INDOUBT
```

If there are any in-doubt messages, you must resolve them before proceeding. For example, you might try issuing the RESOLVE channel command or stopping and restarting the channel.

6. Delete the local queue.

   When you are satisfied that there are no more messages to be delivered to the inventory application at TORONTO, you can delete the queue:

   ```
   DELETE QLOCAL(INVENTQ)
   ```
7. You can now remove the inventory application from the system in Toronto Removing the application avoids duplication and saves space on the system.

Results

The cluster set up by this task is like that setup by the previous task. The difference is the INVENTQ queue is no longer available at queue manager TORONTO.

When you took the queue out of service in step 1, the TORONTO queue manager sent a message to the two full repository queue managers. It notified them of the change in status. The full repository queue managers pass on this information to other queue managers in the cluster that have requested updates to information concerning the INVENTQ.

When a queue manager puts a message on the INVENTQ queue the updated partial repository indicates that the INVENTQ queue is available only at the NEWYORK queue manager. The message is sent to the NEWYORK queue manager.

What to do next

In this task, there was only one queue to remove and only one cluster to remove it from.

Suppose that there are many queues referring to a namelist containing many cluster names. For example, the TORONTO queue manager might host not only the INVENTQ, but also the PAYROLLQ, SALESQ, and PURCHASESQ. TORONTO makes these queues available in all the appropriate clusters, INVENTORY, PAYROLL, SALES, and PURCHASES . Define a namelist of the cluster names on the TORONTO queue manager:

```
DEFINE NAMELIST(TOROLIST)
       DESCR('List of clusters TORONTO is in')
       NAMES(INVENTORY, PAYROLL, SALES, PURCHASES)
```

Add the namelist to each queue definition:

```
DEFINE QLOCAL(INVENTQ) CLUSNL(TOROLIST)
DEFINE QLOCAL(PAYROLLQ) CLUSNL(TOROLIST)
DEFINE QLOCAL(SALESQ) CLUSNL(TOROLIST)
DEFINE QLOCAL(PURCHASESQ) CLUSNL(TOROLIST)
```

Now suppose that you want to remove all those queues from the SALES cluster, because the SALES operation is to be taken over by the PURCHASES operation. All you need to do is alter the TOROLIST namelist to remove the name of the SALES cluster from it.

If you want to remove a single queue from one of the clusters in the namelist, create a namelist, containing the remaining list of cluster names. Then alter the queue definition to use the new namelist. To remove the PAYROLLQ from the INVENTORY cluster:

1. Create a namelist:

```
DEFINE NAMELIST(TOROSHORTLIST)
       DESCR('List of clusters TORONTO is in other than INVENTORY')
       NAMES(PAYROLL, SALES, PURCHASES)
```

2. Alter the PAYROLLQ queue definition:

   ```
   ALTER QLOCAL(PAYROLLQ) CLUSNL(TOROSHORTLIST)
   ```

Moving a full repository to another queue manager

Move a full repository from one queue manager to another, building up the new repository from information held at the second repository.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster has been set up as described in "Adding a queue manager to a cluster" on page 924.
- For business reasons you now want to remove the full repository from queue manager LONDON, and replace it with a full repository at queue manager PARIS. The NEWYORK queue manager is to continue holding a full repository.

About this task

Follow these steps to move a full repository to another queue manager.

Procedure

1. Alter PARIS to make it a full repository queue manager.

   On PARIS, issue the following command:

   ```
   ALTER QMGR REPOS(INVENTORY)
   ```

2. Add a CLUSSDR channel on PARIS

   PARIS currently has a cluster-sender channel pointing to LONDON. LONDON is no longer to hold a full repository for the cluster. PARIS must have a new cluster-sender channel that points to NEWYORK, where the other full repository is now held.

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-sender channel from PARIS to repository at NEWYORK')
   ```

3. Define a CLUSSDR channel on NEWYORK that points to PARIS

   Currently NEWYORK has a cluster-sender channel pointing to LONDON. Now that the other full repository has moved to PARIS, you need to add a new cluster-sender channel at NEWYORK that points to PARIS.

   ```
   DEFINE CHANNEL(INVENTORY.PARIS) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(PARIS.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-sender channel from NEWYORK to repository at PARIS')
   ```

   When you add the cluster-sender channel to PARIS, PARIS learns about the cluster from NEWYORK. It builds up its own full repository using the information from NEWYORK.

4. Check that queue manager PARIS now has a full repository

   Check that queue manager PARIS has built its own full repository from the full repository on queue manager NEWYORK. Issue the following commands:

   ```
   DIS QCLUSTER(*) CLUSTER (INVENTORY)
   DIS CLUSQMGR(*) CLUSTER (INVENTORY)
   ```

   Check that these commands show details of the same resources in this cluster as on NEWYORK.

Note: If queue manager NEWYORK is not available, this building of information cannot complete. Do not move on to the next step until the task is complete.

5. Alter the queue-manager definition on LONDON

   Finally alter the queue manager at LONDON so that it no longer holds a full repository for the cluster. On LONDON, issue the command:

   ```
   ALTER QMGR REPOS(' ')
   ```

   The queue manager no longer receives any cluster information. After 30 days the information that is stored in its full repository expires. The queue manager LONDON now builds up its own partial repository.

6. Remove or change any outstanding definitions.

   When you are sure that the new arrangement of your cluster is working as expected, remove or change manually defined CLUSSDR definitions that are no longer correct.

   - On the PARIS queue manager, you must stop and delete the cluster-sender channel to LONDON, and then issue the start channel command so that the cluster can use the automatic channels again:

     ```
     STOP CHANNEL(INVENTORY.LONDON)
     DELETE CHANNEL(INVENTORY.LONDON)
     START CHANNEL(INVENTORY.LONDON)
     ```

   - On the NEWYORK queue manager, you must stop and delete the cluster-sender channel to LONDON, and then issue the start channel command so that the cluster can use the automatic channels again:

     ```
     STOP CHANNEL(INVENTORY.LONDON)
     DELETE CHANNEL(INVENTORY.LONDON)
     START CHANNEL(INVENTORY.LONDON)
     ```

   - Replace all other cluster-sender channels in the cluster that point to LONDON with channels that point to either NEWYORK or PARIS. In this small example, there are no others. To check whether there are any others that you have forgotten, issue the DISPLAY CHANNEL command from each queue manager, specifying TYPE(CLUSSDR). For example:

     ```
     DISPLAY CHANNEL(*) TYPE(CLUSSDR)
     ```

   It is important that you perform this task as soon as possible after moving the full repository from LONDON to PARIS. In the time before you perform this task, queue managers that have manually defined CLUSSDR channels named INVENTORY.LONDON might send requests for information using this channel.

   After LONDON has ceased to be a full repository, if it receives such requests it will write error messages to its queue manager error log. The following examples show which error messages might be seen on LONDON:

   - `AMQ9428: Unexpected publication of a cluster queue object received`
   - `AMQ9432: Query received by a non-repository queue manager`

   The queue manager LONDON does not respond to the requests for information because it is no longer a full repository. The queue managers requesting information from LONDON must rely on NEWYORK for cluster information until their manually defined CLUSSDR definitions are corrected to point to PARIS. This situation must not be tolerated as a valid configuration in the long term.

Results

Figure 147 on page 960 shows the cluster set up by this task.

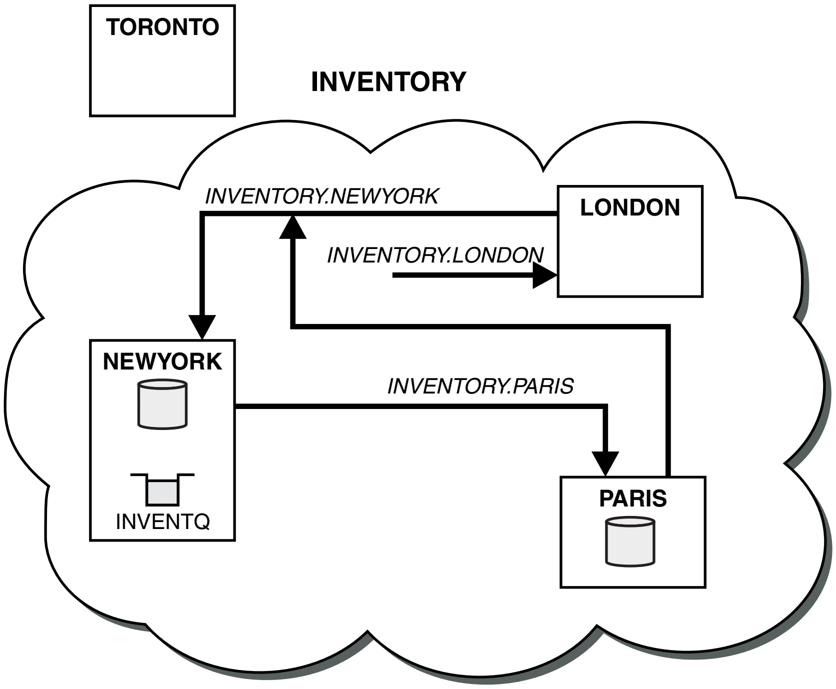


Figure 147. The `INVENTORY` cluster with the full repository moved to `PARIS`

Converting an existing network into a cluster

Convert an existed distributed queuing network to a cluster and add an additional queue manager to increase capacity.

Before you begin

In “Setting up a new cluster” on page 913 through “Moving a full repository to another queue manager” on page 958 you created and extended a new cluster. The next two tasks explore a different approach: that of converting an existing network of queue managers into a cluster.

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- A IBM WebSphere MQ network is already in place, connecting the nationwide branches of a chain store. It has a hub and spoke structure: all the queue managers are connected to one central queue manager. The central queue manager is on the system on which the inventory application runs. The application is driven by the arrival of messages on the `INVENTQ` queue, for which each queue manager has a remote-queue definition.

  This network is illustrated in Figure 148 on page 961.

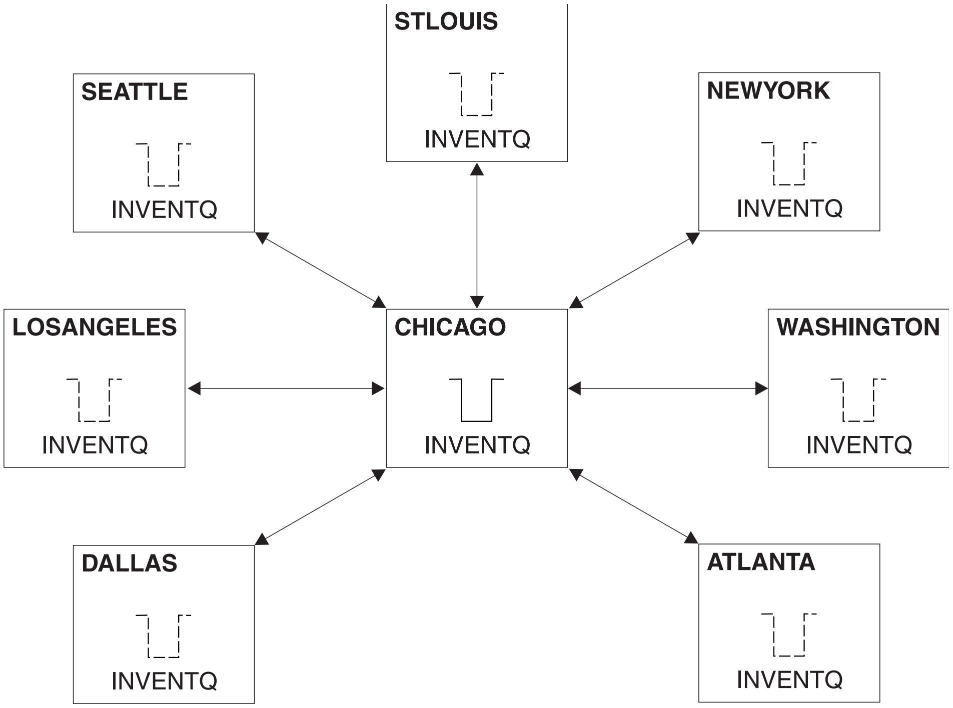


Figure 148. A hub and spoke network

- To ease administration you are going to convert this network into a cluster and create another queue manager at the central site to share the workload.

  The cluster name is CHNSTORE.

  Note: The cluster name CHNSTORE was selected to allow cluster-receiver channel names to be created using names in the format *cluster-name.queue-manager* that do not exceed the maximum length of 20 characters, for example CHNSTORE.WASHINGTON.
- Both the central queue managers are to host full repositories and be accessible to the inventory application.
- The inventory application is to be driven by the arrival of messages on the INVENTQ queue hosted by either of the central queue managers.
- The inventory application is to be the only application running in parallel and accessible by more than one queue manager. All other applications continue to run as before.
- All the branches have network connectivity to the two central queue managers.
- The network protocol is TCP.

About this task

Follow these steps to convert an existing network into a cluster.

Procedure

1. Review the inventory application for message affinities.

   Before proceeding ensure that the application can handle message affinities. Message affinities are the relationship between conversational messages that are exchanged between two applications, where the messages must be processed by a particular queue manager or in a particular sequence. For more information on message affinities, see: “Handling message affinities” on page 1000
2. Alter the two central queue managers to make them full repository queue managers.

The two queue managers CHICAGO and CHICAGO2 are at the hub of this network. You have decided to concentrate all activity associated with the chain store cluster on to those two queue managers. As well as the inventory application and the definitions for the INVENTQ queue, you want these queue managers to host the two full repositories for the cluster. At each of the two queue managers, issue the following command:

```
ALTER QMGR REPOS(CHNSTORE)
```

3. Define a CLUSRCVR channel on each queue manager.

   At each queue manager in the cluster, define a cluster-receiver channel and a cluster-sender channel. It does not matter which channel you define first.

   Make a CLUSRCVR definition to advertise each queue manager, its network address, and other information, to the cluster. For example, on queue manager ATLANTA:

   ```
   DEFINE CHANNEL(CHNSTORE.ATLANTA) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(ATLANTA.CHSTORE.COM) CLUSTER(CHNSTORE)
   DESCR('Cluster-receiver channel')
   ```

4. Define a CLUSSDR channel on each queue manager

   Make a CLUSSDR definition at each queue manager to link that queue manager to one or other of the full repository queue managers. For example, you might link ATLANTA to CHICAGO2:

   ```
   DEFINE CHANNEL(CHNSTORE.CHICAGO2) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(CHICAGO2.CHSTORE.COM) CLUSTER(CHNSTORE)
   DESCR('Cluster-sender channel to repository queue manager')
   ```

5. Install the inventory application on CHICAGO2.

   You already have the inventory application on queue manager CHICAGO. Now you need to make a copy of this application on queue manager CHICAGO2.

6. Define the INVENTQ queue on the central queue managers.

   On CHICAGO, modify the local queue definition for the queue INVENTQ to make the queue available to the cluster. Issue the command:

   ```
   ALTER QLOCAL(INVENTQ) CLUSTER(CHNSTORE)
   ```

   On CHICAGO2, make a definition for the same queue:

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(CHNSTORE)
   ```

   On z/OS, you can use the MAKEDEF option of the COMMAND function of **CSQUTIL** to make an exact copy on CHICAGO2 of the INVENTQ on CHICAGO.

   When you make these definitions, a message is sent to the full repositories at CHICAGO and CHICAGO2 and the information in them is updated. The queue manager finds out from the full repositories when it puts a message to the INVENTQ, that there is a choice of destinations for the messages.

7. Check that the cluster changes have been propagated.

   Check that the definitions you created in the previous step have been propagated though the cluster. Issue the following command on a full repository queue manager:

   ```
   DIS QCLUSTER(INVENTQ)
   ```

Adding a new, interconnected cluster

Add a new cluster that shares some queue managers with an existing cluster.

Before you begin

Note:

1. For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.
2. Before starting this task, check for queue-name clashes and understand the consequences. You might need to rename a queue, or set up queue aliases before you can proceed.

Scenario:

- A WebSphere MQ cluster has been set up as described in “Converting an existing network into a cluster” on page 960 .
- A new cluster called MAILORDER is to be implemented. This cluster comprises four of the queue managers that are in the CHNSTORE cluster; CHICAGO, CHICAGO2, SEATTLE, and ATLANTA, and two additional queue managers; HARTFORD and OMAHA . The MAILORDER application runs on the system at Omaha, connected to queue manager OMAHA. It is driven by the other queue managers in the cluster putting messages on the MORDERQ queue.
- The full repositories for the MAILORDER cluster are maintained on the two queue managers CHICAGO and CHICAGO2.
- The network protocol is TCP.

About this task

Follow these steps to add a new, interconnected cluster.

Procedure

1. Create a namelist of the cluster names.

   The full repository queue managers at CHICAGO and CHICAGO2 are now going to hold the full repositories for both of the clusters CHNSTORE and MAILORDER . First, create a namelist containing the names of the clusters. Define the namelist on CHICAGO and CHICAGO2, as follows:

   ```
   DEFINE NAMELIST(CHAINMAIL)
          DESCR('List of cluster names')
          NAMES(CHNSTORE, MAILORDER)
   ```
2. Alter the two queue-manager definitions.

   Now alter the two queue-manager definitions at CHICAGO and CHICAGO2. Currently these definitions show that the queue managers hold full repositories for the cluster CHNSTORE. Change that definition to show that the queue managers hold full repositories for all clusters listed in the CHAINMAIL namelist. Alter the CHICAGO and CHICAGO2 queue manager definitions:

   ```
   ALTER QMGR REPOS(' ') REPOSNL(CHAINMAIL)
   ```
3. Alter the CLUSRCVR channels on CHICAGO and CHICAGO2.

   The CLUSRCVR channel definitions at CHICAGO and CHICAGO2 show that the channels are available in the cluster CHNSTORE. You need to change the cluster-receiver definition to show that the channels are available to all clusters listed in the CHAINMAIL namelist. Change the cluster-receiver definition at CHICAGO:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO) CHLTYPE(CLUSRCVR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```

   At CHICAGO2, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO2) CHLTYPE(CLUSRCVR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```
4. Alter the CLUSSDR channels on CHICAGO and CHICAGO2.

   Change the two CLUSSDR channel definitions to add the namelist. At CHICAGO, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO2) CHLTYPE(CLUSSDR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```

   At CHICAGO2, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO) CHLTYPE(CLUSSDR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```
5. Create a namelist on SEATTLE and ATLANTA.

   Because SEATTLE and ATLANTA are going to be members of more than one cluster, you must create a namelist containing the names of the clusters. Define the namelist on SEATTLE and ATLANTA, as follows:

```
DEFINE NAMELIST(CHAINMAIL)
       DESCR('List of cluster names')
       NAMES(CHNSTORE, MAILORDER)
```

6. Alter the CLUSRCVR channels on SEATTLE and ATLANTA.

   The CLUSRCVR channel definitions at SEATTLE and ATLANTA show that the channels are available in the cluster CHNSTORE. Change the cluster-receive channel definitions to show that the channels are available to all clusters listed in the CHAINMAIL namelist. At SEATTLE, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.SEATTLE) CHLTYPE(CLUSRCVR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```

   At ATLANTA, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.ATLANTA) CHLTYPE(CLUSRCVR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```

7. Alter the CLUSSDR channels on SEATTLE and ATLANTA.

   Change the two CLUSSDR channel definitions to add the namelist. At SEATTLE, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO) CHLTYPE(CLUSSDR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```

   At ATLANTA, enter the command:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO2) CHLTYPE(CLUSSDR)
         CLUSTER(' ') CLUSNL(CHAINMAIL)
   ```

8. Define CLUSRCVR and CLUSSDR channels on HARTFORD and OMAHA .

   On the two new queue managers HARTFORD and OMAHA, define cluster-receiver and cluster-sender channels. It does not matter in which sequence you make the definitions. At HARTFORD, enter:

   ```
   DEFINE CHANNEL(MAILORDER.HARTFORD) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(HARTFORD.CHSTORE.COM) CLUSTER(MAILORDER)
   DESCR('Cluster-receiver channel for HARTFORD')

   DEFINE CHANNEL(MAILORDER.CHICAGO) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(CHICAGO.CHSTORE.COM) CLUSTER(MAILORDER)
   DESCR('Cluster-sender channel from HARTFORD to repository at CHICAGO')
   ```

   At OMAHA, enter:

   ```
   DEFINE CHANNEL(MAILORDER.OMAHA) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(OMAHA.CHSTORE.COM) CLUSTER(MAILORDER)
   DESCR('Cluster-receiver channel for OMAHA')

   DEFINE CHANNEL(MAILORDER.CHICAGO) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(CHICAGO.CHSTORE.COM) CLUSTER(MAILORDER)
   DESCR('Cluster-sender channel from OMAHA to repository at CHICAGO')
   ```

9. Define the MORDERQ queue on OMAHA .

   The final step to complete this task is to define the queue MORDERQ on the queue manager OMAHA . At OMAHA, enter:

   ```
   DEFINE QLOCAL(MORDERQ) CLUSTER(MAILORDER)
   ```

10. Check that the cluster changes have been propagated.

    Check that the definitions you created with the previous steps have been propagated though the cluster. Issue the following commands on a full repository queue manager:

    ```
    DIS QCLUSTER (MORDERQ)
    DIS CLUSQMGR
    ```

11.

Results

The cluster set up by this task is shown in Figure 149 on page 965.

Now we have two overlapping clusters. The full repositories for both clusters are held at CHICAGO and CHICAGO2. The mail order application that runs on OMAHA is independent of the inventory application that runs at CHICAGO. However, some of the queue managers that are in the CHNSTORE cluster are also in the MAILORDER cluster, and so they can send messages to either application. Before carrying out this task to overlap two clusters, be aware of the possibility of queue-name clashes.

Suppose that on NEWYORK in cluster CHNSTORE and on OMAHA in cluster MAILORDER, there is a queue called ACCOUNTQ . If you overlap the clusters and then an application on SEATTLE puts a message to the queue ACCOUNTQ, the message can go to either instance of the ACCOUNTQ .

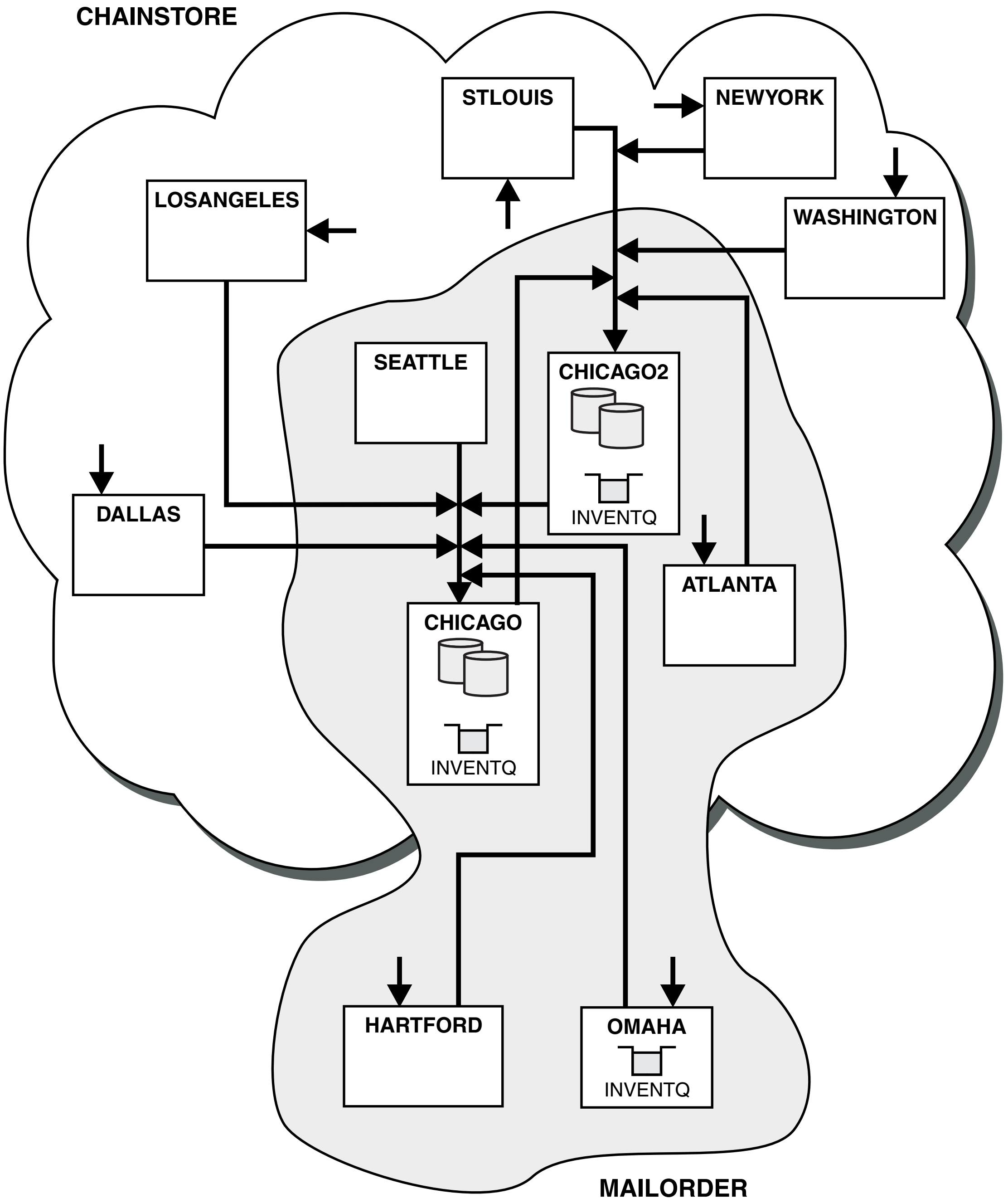


Figure 149. Interconnected clusters

What to do next

Suppose you decide to merge the MAILORDER cluster with the CHNSTORE cluster to form one large cluster called CHNSTORE.

To merge the MAILORDER cluster with the CHNSTORE cluster, such that CHICAGO and CHICAGO2 hold the full repositories:

- Alter the queue manager definitions for CHICAGO and CHICAGO2, removing the REPOSNL attribute, which specifies the namelist (CHAINMAIL), and replacing it with a REPOS attribute specifying the cluster name (CHNSTORE). For example:
  ```
  ALTER QMGR(CHICAGO) REPOSNL(' ') REPOS(CHNSTORE)
  ```
- On each queue manager in the MAILORDER cluster, alter all the channel definitions and queue definitions to change the value of the CLUSTER attribute from MAILORDER to CHNSTORE. For example, at HARTFORD, enter:
  ```
  ALTER CHANNEL(MAILORDER.HARTFORD) CLUSTER(CHNSTORE)
  ```

  At OMAHA enter:
  ```
  ALTER QLOCAL(MORDERQ) CLUSTER(CHNSTORE)
  ```
- Alter all definitions that specify the cluster namelist CHAINMAIL, that is, the CLUSRCVR and CLUSSDR channel definitions at CHICAGO, CHICAGO2, SEATTLE, and ATLANTA, to specify instead the cluster CHNSTORE.

From this example, you can see the advantage of using namelists. Instead of altering the queue manager definitions for CHICAGO and CHICAGO2 you can alter the value of the namelist CHAINMAIL. Similarly, instead of altering the CLUSRCVR and CLUSSDR channel definitions at CHICAGO, CHICAGO2, SEATTLE, and ATLANTA, you can achieve the required result by altering the namelist.

Removing a cluster network

Remove a cluster from a network and restore the distributed queuing configuration.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- A IBM WebSphere MQ cluster has been set up as described in "Converting an existing network into a cluster" on page 960.
- This cluster is now to be removed from the system. The network of queue managers is to continue functioning as it did before the cluster was implemented.

About this task

Follow these steps to remove a cluster network.

Procedure

1. Remove cluster queues from the CHNSTORE cluster.

   On both CHICAGO and CHICAGO2, modify the local queue definition for the queue INVENTQ to remove the queue from the cluster. Issue the command:
   ```
   ALTER QLOCAL(INVENTQ) CLUSTER(' ')
   ```
   When you alter the queue, the information in the full repositories is updated and propagated throughout the cluster. Active applications using MQOO_BIND_NOT_FIXED, and applications using MQOO_BIND_AS_Q_DEF where the queue has been defined with DEFBIND(NOTFIXED), fail on the next attempted MQPUT or MQPUT1 call. The reason code MQRC_UNKNOWN_OBJECT_NAME is returned.

You do not have to perform Step 1 first, but if you do not, perform it instead after Step 4.

2. Stop all applications that have access to cluster queue.

   Stop all applications that have access to cluster queues. If you do not, some cluster information might remain on the local queue manager when you refresh the cluster in Step 5. This information is removed when all applications have stopped and the cluster channels have disconnected.

3. Remove the repository attribute from the full repository queue managers.

   On both CHICAGO and CHICAGO2, modify the queue manager definitions to remove the repository attribute. To do this issue the command:

   ```
   ALTER QMGR REPOS(' ')
   ```

   The queue managers inform the other queue managers in the cluster that they no longer hold the full repositories. When the other queue managers receive this information, you see a message indicating that the full repository has ended. You also see one or more messages indicating that there are no longer any repositories available for the cluster CHNSTORE .

4. Remove cluster channels.

   On CHICAGO remove the cluster channels:

   ```
   ALTER CHANNEL(CHNSTORE.CHICAGO2) CHLTYPE(CLUSSDR) CLUSTER(' ')
   ALTER CHANNEL(CHNSTORE.CHICAGO) CHLTYPE(CLUSRCVR) CLUSTER(' ')
   ```

   Note: It is important to issue the CLUSSDR command first, then CLUSRCVR command. Do not issue the CLUSRCVR command first, then the CLUSSDR command. Doing so, creates indoubt channels that have a STOPPED status. You then need to issue a START CHANNEL command to recover the stopped channels; for example, START CHANNEL(CHNSTORE.CHICAGO).

   You see messages indicating that there are no repositories for the cluster CHNSTORE.

   If you did not remove the cluster queues as described in Step 1, do so now.

5. Stop cluster channels.

   On CHICAGO stop the cluster channels with the following commands:

   ```
   STOP CHANNEL(CHNSTORE.CHICAGO2)
   STOP CHANNEL(CHNSTORE.CHICAGO)
   ```

6. Repeat steps 4 and 5 for each queue manager in the cluster.

7. Stop the cluster channels, then remove all definitions for the cluster channels and cluster queues from each queue manager.

8. Optional: Clear the cached cluster information held by the queue manager. Although the queue managers are no longer members of the cluster, they each retain a cached copy of information about the cluster. If you want to remove this data, see task “Restoring a queue manager to its pre-cluster state” on page 971.

9. Replace the remote-queue definitions for the INVENTQ

   So that the network can continue to function, replace the remote queue definition for the INVENTQ at every queue manager.

10. Tidy up the cluster.

    Delete any queue or channel definitions no longer required.

Removing a queue manager from a cluster

Remove a queue manager from a cluster, in scenarios where the queue manager can communicate normally with at least one full repository in the cluster.

Before you begin

This method is the best practice for scenarios in which at least one full repository is available, and can be contacted by the queue manager that is being removed. This method involves the least manual intervention, and allows the queue manager to negotiate a controlled withdrawal from the cluster. If the queue manager that is being removed cannot contact a full repository, see "Removing a queue manager from a cluster: Alternative method" on page 969.

Before you remove the queue manager from the cluster, you must ensure that the queue manager is no longer hosting resources that are needed by the cluster:

- If the queue manager hosts a full repository, complete steps 1-4 from "Moving a full repository to another queue manager" on page 958.
- If the queue manager hosts cluster queues, complete steps 1-7 from "Removing a cluster queue from a queue manager" on page 956.
- If the queue manager hosts cluster topics, either delete the topics (for example by using the DELETE TOPIC command), or move them to other hosts.

  Note: If you remove a queue manager from a cluster, and the queue manager still hosts a cluster topic, then the queue manager might continue to attempt to deliver publications to the queue managers that are left in the cluster until the topic is deleted.

About this task

This example task removes the queue manager `LONDON` from the `INVENTORY` cluster. The `INVENTORY` cluster is set up as described in "Adding a queue manager to a cluster" on page 924, and modified as described in "Removing a cluster queue from a queue manager" on page 956.

The process for removing a queue manager from a cluster is more complicated than the process of adding a queue manager.

When a queue manager joins a cluster, the existing members of the cluster have no knowledge of the new queue manager and so have no interactions with it. New sender and receiver channels must be created on the joining queue manager so that it can connect to a full repository.

When a queue manager is removed from a cluster, it is likely that applications connected to the queue manager are using objects such as queues that are hosted elsewhere in the cluster. Also, applications that are connected to other queue managers in the cluster might be using objects hosted on the target queue manager. As a result of these applications, the current queue manager might create additional sender channels to establish communication with cluster members other than the full repository that it used to join the cluster. Every queue manager in the cluster has a cached copy of data that describes other cluster members. This might include the one that is being removed.

Procedure

1. Alter the manually defined cluster receiver channels to remove them from the cluster, on queue manager `LONDON`:
   ```
   ALTER CHANNEL(INVENTORY.LONDON) CHLTYPE(CLUSRCVR) CLUSTER(' ')
   ```
2. Alter the manually defined cluster sender channels to remove them from the cluster, on queue manager `LONDON`:
   ```
   ALTER CHANNEL(INVENTORY.PARIS) CHLTYPE(CLUSSDR) CLUSTER(' ')
   ```

The other queue managers in the cluster learn that this queue manager and its cluster resources are no longer part of the cluster.

3. Monitor the cluster transmit queue, on queue manager LONDON, until there are no messages that are waiting to flow to any full repository in the cluster.

   ```
   DISPLAY CHSTATUS(INVENTORY.LONDON) XQMSGSA
   ```

   If messages remain on the transmit queue, determine why they are not being sent to the PARIS and NEWYORK full repositories before continuing.

Results

The queue manager LONDON is no longer part of the cluster. However, it can still function as an independent queue manager.

What to do next

The result of these changes can be confirmed by issuing the following command on the remaining members of the cluster:

```
DISPLAY CLUSQMGR(LONDON)
```

The queue manager continues to be displayed until the auto-defined cluster sender channels to it have stopped. You can wait for this to happen, or, continue to monitor for active instances by issuing the following command:

```
DISPLAY CHANNEL(INVENTORY.LONDON)
```

When you are confident that no more messages are being delivered to this queue manager, you can stop the cluster sender channels to LONDON by issuing the following command on the remaining members of the cluster:

```
STOP CHANNEL(INVENTORY.LONDON) STATUS(INACTIVE)
```

After the changes are propagated throughout the cluster, and no more messages are being delivered to this queue manager, stop and delete the CLUSRCVR channel on LONDON:

```
STOP CHANNEL(INVENTORY.LONDON)
DELETE CHANNEL(INVENTORY.LONDON)
```

The removed queue manager can be added back into the cluster at a later point as described in “Adding a queue manager to a cluster” on page 924. The removed queue manager continues to cache knowledge of the remaining members of the cluster for up to 90 days. If you prefer not to wait until this cache expires, it can be forcibly removed as described in “Restoring a queue manager to its pre-cluster state” on page 971.

Removing a queue manager from a cluster: Alternative method:

Remove a queue manager from a cluster, in scenarios where, because of a significant system or configuration issue, the queue manager cannot communicate with any full repository in the cluster.

Before you begin

This alternative method of removing a queue manager from a cluster manually stops and deletes all cluster channels linking the removed queue manager to the cluster, and forcibly removes the queue manager from the cluster. This method is used in scenarios where the queue manager that is being removed cannot communicate with any of the full repositories. This might be (for example) because the queue manager has stopped working, or because there has been a prolonged communications failure between the queue manager and the cluster. Otherwise,use the most common method: “Removing a queue manager from a cluster” on page 968.

Before you remove the queue manager from the cluster, you must ensure that the queue manager is no longer hosting resources that are needed by the cluster:

- If the queue manager hosts a full repository, complete steps 1-4 from “Moving a full repository to another queue manager” on page 958.
- If the queue manager hosts cluster queues, complete steps 1-7 from “Removing a cluster queue from a queue manager” on page 956.
- If the queue manager hosts cluster topics, either delete the topics (for example by using the DELETE TOPIC command), or move them to other hosts.

  Note: If you remove a queue manager from a cluster, and the queue manager still hosts a cluster topic, then the queue manager might continue to attempt to deliver publications to the queue managers that are left in the cluster until the topic is deleted.

About this task

This example task removes the queue manager `LONDON` from the `INVENTORY` cluster. The `INVENTORY` cluster is set up as described in “Adding a queue manager to a cluster” on page 924, and modified as described in “Removing a cluster queue from a queue manager” on page 956.

The process for removing a queue manager from a cluster is more complicated than the process of adding a queue manager.

When a queue manager joins a cluster, the existing members of the cluster have no knowledge of the new queue manager and so have no interactions with it. New sender and receiver channels must be created on the joining queue manager so that it can connect to a full repository.

When a queue manager is removed from a cluster, it is likely that applications connected to the queue manager are using objects such as queues that are hosted elsewhere in the cluster. Also, applications that are connected to other queue managers in the cluster might be using objects hosted on the target queue manager. As a result of these applications, the current queue manager might create additional sender channels to establish communication with cluster members other than the full repository that it used to join the cluster. Every queue manager in the cluster has a cached copy of data that describes other cluster members. This might include the one that is being removed.

This procedure might be appropriate in an emergency, when it is not possible to wait for the queue manager to leave the cluster gracefully.

Procedure

1. Stop all channels used to communicate with other queue managers in the cluster. Use `MODE(FORCE)` to stop the `CLUSRCVR` channel, on queue manager `LONDON`. Otherwise you might need to wait for the sender queue manager to stop the channel:

   ```
   STOP CHANNEL(INVENTORY.LONDON) MODE(FORCE)
   STOP CHANNEL(INVENTORY.TORONTO)
   STOP CHANNEL(INVENTORY.PARIS)
   STOP CHANNEL(INVENTORY.NEWYORK)
   ```

2. Monitor the channel states, on queue manager `LONDON`, until the channels stop:

   ```
   DISPLAY CHSTATUS(INVENTORY.LONDON)
   DISPLAY CHSTATUS(INVENTORY.TORONTO)
   DISPLAY CHSTATUS(INVENTORY.PARIS)
   DISPLAY CHSTATUS(INVENTORY.NEWYORK)
   ```

   No more application messages are sent to or from the other queue managers in the cluster after the channels stop.

3. Delete the manually defined cluster channels, on queue manager `LONDON`:

```
DELETE CHANNEL(INVENTORY.NEWYORK)
DELETE CHANNEL(INVENTORY.TORONTO)
```

4. The remaining queue managers in the cluster still retain knowledge of the removed queue manager, and might continue to send messages to it. To purge the knowledge from the remaining queue managers, reset the removed queue manager from the cluster on one of the full repositories:

   ```
   RESET CLUSTER(INVENTORY) ACTION(FORCEREMOVE) QMNAME(LONDON) QUEUES(YES)
   ```

   If there might be another queue manager in the cluster that has the same name as the removed queue manager, specify the **QMID** of the removed queue manager.

Results

The queue manager LONDON is no longer part of the cluster. However, it can still function as an independent queue manager.

What to do next

The result of these changes can be confirmed by issuing the following command on the remaining members of the cluster:

```
DISPLAY CLUSQMGR(LONDON)
```

The queue manager continues to be displayed until the auto-defined cluster sender channels to it have stopped. You can wait for this to happen, or, continue to monitor for active instances by issuing the following command:

```
DISPLAY CHANNEL(INVENTORY.LONDON)
```

After the changes are propagated throughout the cluster, and no more messages are being delivered to this queue manager, delete the CLUSRCVR channel on LONDON:

```
DELETE CHANNEL(INVENTORY.LONDON)
```

The removed queue manager can be added back into the cluster at a later point as described in “Adding a queue manager to a cluster” on page 924. The removed queue manager continues to cache knowledge of the remaining members of the cluster for up to 90 days. If you prefer not to wait until this cache expires, it can be forcibly removed as described in “Restoring a queue manager to its pre-cluster state.”

Restoring a queue manager to its pre-cluster state

When a queue manager is removed from a cluster, it retains knowledge of the remaining cluster members. This knowledge eventually expires and is deleted automatically. However, if you prefer to delete it immediately, you can use the steps in this topic.

Before you begin

It is assumed that the queue manager has been removed from the cluster, and is no longer performing any work in the cluster. For example, its queues are no longer receiving messages from the cluster, and no applications are waiting for messages to arrive in these queues.

About this task

When a queue manager is removed from a cluster, it retains knowledge of the remaining cluster members for up to 90 days. This can have system benefits, particularly if the queue manager quickly rejoins the cluster. When this knowledge eventually expires, it is deleted automatically. However, there are reasons why you might prefer to delete this information manually. For example:

- You might want to confirm that you have stopped every application on this queue manager that previously used cluster resources. Until the knowledge of the remaining cluster members expires, any such application continues to write to a transmit queue. After the cluster knowledge is deleted, the system generates an error message when such an application tries to use cluster resources.

- When you display status information for the queue manager, you might prefer not to see expiring information about remaining cluster members.

This task uses the INVENTORY cluster as an example. The LONDON queue manager has been removed from the INVENTORY cluster as described in “Removing a queue manager from a cluster” on page 968. To delete knowledge of the remaining members of the cluster, issue the following commands on the LONDON queue manager.

Procedure

1. Remove all memory of the other queue managers in the cluster from this queue manager:

   ```
   REFRESH CLUSTER(INVENTORY) REPOS(YES)
   ```

2. Monitor the queue manager until all the cluster resources are gone:

   ```
   DISPLAY CLUSQMGR(*) CLUSTER(INVENTORY)
   DISPLAY QCLUSTER(*) CLUSTER(INVENTORY)
   DISPLAY TOPIC(*) CLUSTER(INVENTORY)
   ```

Related concepts:

“Comparison of clustering and distributed queuing” on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

“Components of a cluster” on page 890
Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

“Managing IBM WebSphere MQ clusters” on page 912
You can create, extend, and maintain IBM WebSphere MQ clusters.

Related information:

Clusters

Maintaining a queue manager

Suspend and resume a queue manager from a cluster to perform maintenance.

About this task

From time to time, you might need to perform maintenance on a queue manager that is part of a cluster. For example, you might need to take backups of the data in its queues, or apply fixes to the software. If the queue manager hosts any queues, its activities must be suspended. When the maintenance is complete, its activities can be resumed.

Procedure

1. Suspend a queue manager, by issuing the SUSPEND QMGR **runmqsc** command:

   ```
   SUSPEND QMGR CLUSTER(SALES)
   ```

   The SUSPEND **runmqsc** command notifies the queue managers in the SALES cluster that this queue manager has been suspended.

   The purpose of the SUSPEND QMGR command is only to advise other queue managers to avoid sending messages to this queue manager if possible. It does not mean that the queue manager is disabled. Some messages that have to be handled by this queue manager are still sent to it, for example when this queue manager is the only host of a clustered queue.

   While the queue manager is suspended the workload management routines avoid sending messages to it. Messages that have to be handled by that queue manager include messages sent by the local queue manager.

   WebSphere MQ uses a workload balancing algorithm to determine which destinations are suitable, rather than selecting the local queue manager whenever possible.

   a. Enforce the suspension of a queue manager by using the FORCE option on the SUSPEND QMGR command:

```
SUSPEND QMGR CLUSTER(SALES) MODE(FORCE)
```

   MODE(FORCE) forcibly stops all inbound channels from other queue managers in the cluster. If you do not specify MODE(FORCE), the default MODE(QUIESCE) applies.

2. Do whatever maintenance tasks are necessary.
3. Resume the queue manager by issuing the RESUME QMGR **runmqsc** command:

```
RESUME QMGR CLUSTER(SALES)
```

Results

The RESUME **runmqsc** command notifies the full repositories that the queue manager is available again. The full repository queue managers disseminate this information to other queue managers that have requested updates to information concerning this queue manager.

Maintaining the cluster transmission queue

Make every effort to keep cluster transmission queues available. They are essential to the performance of clusters.

Before you begin

- Make sure that the cluster transmission queue does not become full.
- Take care not to issue an ALTER **runmqsc** command to set it either get disabled or put disabled accidentally.
- Make sure that the medium the cluster transmission queue is stored on does not become full.

Refreshing a cluster queue manager

You can remove auto-defined channels and auto-defined cluster objects from the local repository using the REFRESH CLUSTER command. No messages are lost.

Before you begin

You might be asked to use the command by your IBM Support Center. Do not use the command without careful consideration. For example, for large clusters use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See “Clustering: Using REFRESH CLUSTER best practices” on page 1030.

About this task

A queue manager can make a fresh start in a cluster. In normal circumstances, you do not need to use the REFRESH CLUSTER command.

Procedure

Issue the REFRESH CLUSTER **MQSC** command from a queue manager to remove auto-defined cluster queue-manager and queue objects from the local repository.
The command only removes objects that refer to other queue managers, it does not remove objects relating to the local queue manager. The command also removes auto-defined channels. It removes channels that do not have messages on the cluster transmission queue and are not attached to a full repository queue manager.

Results

Effectively, the REFRESH CLUSTER command allows a queue manager to be cold-started with respect to its full repository content. WebSphere MQ ensures that no data is lost from your queues.

Related concepts:

"Clustering: Using REFRESH CLUSTER best practices" on page 1030
You use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster. You should not need to use this command, except in exceptional circumstances. If you do need to use it, there are special considerations about how you use it. This information is a guide based on testing and feedback from customers.

Recovering a queue manager

Bring the cluster information about a queue manager up to date using the REFRESH CLUSTER **runmqsc** command. Follow this procedure after recovering a queue manager from a point-in-time backup.

Before you begin

You have restored a cluster queue manager from a point-in-time backup of a linear log.

About this task

To recover a queue manager in a cluster, restore the queue manager, and then bring the cluster information up to date using the REFRESH CLUSTER **runmqsc** command.

Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

Procedure

Issue the REFRESH CLUSTER command on the restored queue manager for all clusters in which the queue manager participates.

What to do next

There is no need to issue the REFRESH CLUSTER command on any other queue manager.

Related concepts:

"Clustering: Using REFRESH CLUSTER best practices" on page 1030
You use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster. You should not need to use this command, except in exceptional circumstances. If you do need to use it, there are special considerations about how you use it. This information is a guide based on testing and feedback from customers.

Configuring cluster channels for availability

Follow good configuration practices to keep cluster channels running smoothly if there are intermittent network stoppages.

Before you begin

Clusters relieve you of the need to define channels, but you still need to maintain them. The same channel technology is used for communication between queue managers in a cluster as is used in distributed queuing. To understand about cluster channels, you need to be familiar with matters such as:

- How channels operate
- How to find their status

- How to use channel exits

About this task

You might want to give some special consideration to the following points:

Procedure

Consider the following points when configuring cluster channels

- Choose values for `HBINT` or `KAINT` on cluster-sender channels and cluster-receiver channels that do not burden the network with lots of heartbeat or keep alive flows. An interval less than about 10 seconds gives false failures, if your network sometimes slows down and introduces delays of this length.
- Set the `BATCHHB` value to reduce the window for causing a marooned message because it is indoubt on a failed channel. An indoubt batch on a failed channel is more likely to occur if the batch is given longer to fill. If the message traffic along the channel is sporadic with long periods of time between bursts of messages a failed batch is more likely.
- A problem arises if the cluster-sender end of a channel fails and then tries to restart before the heartbeat or keep alive has detected the failure. The channel-sender restart is rejected if the cluster-receiver end of the channel has remained active. To avoid the failure, arrange for the cluster-receiver channel to be terminated and restarted when a cluster-sender channel attempts to restart.

  On platforms other than z/OS

  Control the problem of the cluster-receiver end of the channel has remaining active using the `AdoptNewMCA`, `AdoptNewMCATimeout`, and `AdoptNewMCACheck` attributes in the `qm.ini` file or the Windows NT Registry.

Routing messages to and from clusters

Use queue aliases, queue manager aliases, and remote queue definitions to connect clusters to external queue managers and other clusters.

For details on routing messages to and from clusters, see the following subtopics:

Related concepts:

"Comparison of clustering and distributed queuing" on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

"Components of a cluster" on page 890
Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

"Managing IBM WebSphere MQ clusters" on page 912
You can create, extend, and maintain IBM WebSphere MQ clusters.

"Queue-manager aliases and clusters" on page 985
Use queue-manager aliases to hide the name of queue managers when sending messages into or out of a cluster, and to workload balance messages sent to a cluster.

"Queue aliases and clusters" on page 988
Use queue aliases to hide the name of a cluster queue, to cluster a queue, adopt different attributes, or adopt different access controls.

"Reply-to queue aliases and clusters" on page 987
A reply-to queue alias definition is used to specify alternative names for reply information. Reply-to queue alias definitions can be used with clusters just the same as in a distributed queuing environment.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

"Setting up a new cluster" on page 913
Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Related information:

Clusters

How clusters work

Configuring request/reply to a cluster

Configure a request/reply message path from a queue manager outside a cluster. Hide the inner details of the cluster by using a gateway queue manager as the communication path to and from the cluster.

Before you begin

Figure 150 on page 977 shows a queue manager called QM3 that is outside the cluster called DEMO. QM3 could be a queue manager on a WebSphere MQ product that does not support clusters. QM3 hosts a queue called Q3, which is defined as follows:

```
DEFINE QLOCAL(Q3)
```

Inside the cluster are two queue managers called QM1 and QM2. QM2 hosts a cluster queue called Q2, which is defined as follows:

```
DEFINE QLOCAL(Q2) CLUSTER(DEMO)
```

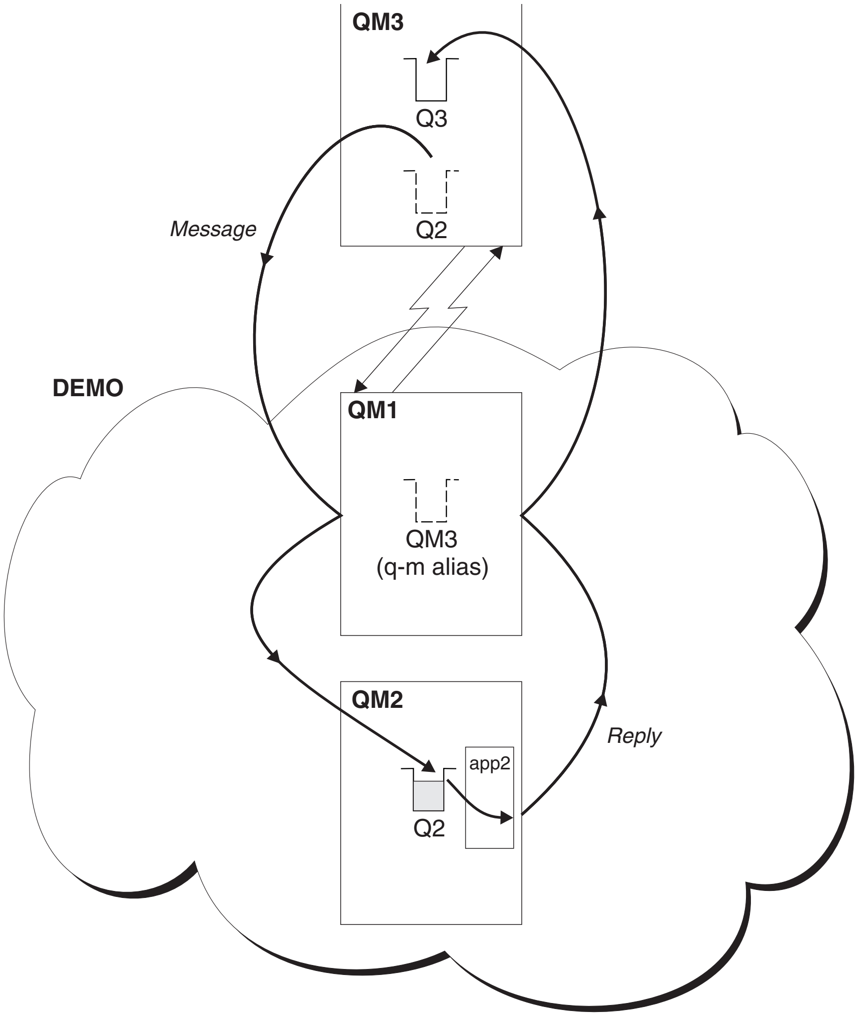


Figure 150. Putting from a queue manager outside the cluster

About this task

Follow the advice in the procedure to set up the path for the request and reply messages.

Procedure

1. Send the request message to the cluster.

   Consider how the queue manager that is outside the cluster puts a message to the queue `Q2` at `QM2`, that is inside the cluster. A queue manager outside the cluster must have a `QREMOTE` definition for each queue in the cluster that it puts messages to.

   a. Define a remote queue for `Q2` on `QM3`.

   ```
   DEFINE QREMOTE(Q2) RNAME(Q2) RQMNAME(QM2) XMITQ(QM1)
   ```

   Because `QM3` is not part of a cluster, it must communicate using distributed queuing techniques. Therefore, it must also have a sender-channel and a transmission queue to `QM1`. `QM1` needs a corresponding receiver channel. The channels and transmission queues are not shown explicitly in Figure 150.

   In the example, an application at `QM3` issues an MQPUT call to put a message to `Q2`. The `QREMOTE` definition causes the message to be routed to `Q2` at `QM2` using the sender-channel that is getting messages from the `QM1` transmission queue.

2. Receive the reply message from the cluster.

Use a queue manager alias to create a return path for replies to a queue manager outside the cluster. The gateway, QM1, advertises a queue-manager alias for the queue manager that is outside the cluster, QM3. It advertises QM3 to the queue managers inside the cluster by adding the cluster attribute to a queue manager alias definition for QM3. A queue manager alias definition is like a remote queue definition, but with a blank RNAME.

a. Define a queue manager alias for QM3 on QM1.

   ```
   DEFINE QREMOTE(QM3) RNAME(' ') RQMNAME(QM3) CLUSTER(DEMO)
   ```

   We must consider the choice of name for the transmission queue used to forward replies back from QM1 to QM3. Implicit in the QREMOTE definition, by the omission of the XMITQ attribute, is the name of the transmission queue is QM3. But QM3 is the same name as we expect to advertise to the rest of the cluster using the queue manager alias. WebSphere MQ does not allow you to give both the transmission queue and the queue manager alias the same name. One solution is to create a transmission queue to forward messages to QM3 with a different name to the queue manager alias.

b. Provide the transmission queue name in the QREMOTE definition.

   ```
   DEFINE QREMOTE(QM3) RNAME(' ') RQMNAME(QM3) CLUSTER(DEMO) XMITQ(QM3.XMIT)
   ```

   The new queue manager alias couples the new transmission queue called QM3.XMIT with the QM3 queue manager alias. It is a simple and correct solution, but not wholly satisfactory. It has broken the naming convention for transmission queues that they are given the same name as the target queue manager. Are there any alternative solutions that preserve the transmission queue naming convention?

   The problem arises because the requester defaulted to passing QM3 as the reply-to queue manager name in the request message that is sent from QM3. The server on QM2 uses the QM3 reply-to queue manager name to address QM3 in its replies. The solution required QM1 to advertise QM3 as the queue manager alias to return reply messages to and prevented QM1 from using QM3 as the name of the transmission queue.

   Instead of defaulting to providing QM3 as the reply-to queue manager name, applications on QM3 need to pass a reply-to queue manager alias to QM1 for reply messages. The gateway queue manager QM1 advertises the queue manager alias for replies to QM3 rather than QM3 itself, avoiding the conflict with the name of the transmission queue.

c. Define a queue manager alias for QM3 on QM1.

   ```
   DEFINE QREMOTE(QM3.ALIAS) RNAME(' ') RQMNAME(QM3) CLUSTER(DEMO)
   ```

   Two changes to the configuration commands are required.

   1) The QREMOTE at QM1 now advertises our queue manager alias QM3.ALIAS to the rest of the cluster, coupling it to the name of the real queue manager QM3. QM3 is again the name of the transmission queue to send reply queues back to QM3
   2) The client application must provide QM3.ALIAS as the name of the reply-to queue manager when it constructs the request message. You can provide QM3.ALIAS to the client application in one of two ways.
      - Code QM3.ALIAS in the reply-to queue manager name field constructed by MQPUT in the MQMD. You must do it this way if you are using a dynamic queue for replies.
      - Use a reply-to queue alias, Q3.ALIAS, rather than a reply-to queue when providing the reply-to queue name.

        ```
        DEFINE QREMOTE(Q3.ALIAS) RNAME(Q3) RQMNAME(QM3.ALIAS)
        ```

What to do next

Note: You cannot demonstrate the use of reply-to queue aliases with **AMQSREQ0**. It opens the reply-to queue using the queue name provided in parameter 3, or the default SYSTEM.SAMPLE.REPLY model queue. You need to modify the sample providing another parameter containing the reply-to queue alias to name the reply-to queue manager alias for MQPUT.

Related tasks:

"Hiding the name of a cluster target queue manager"
Route a message to a cluster queue that is defined on any queue manager in a cluster without naming the queue manager.

Hiding the name of a cluster target queue manager:

Route a message to a cluster queue that is defined on any queue manager in a cluster without naming the queue manager.

Before you begin

- Avoid revealing the names of queue managers that are inside the cluster to queue managers that are outside the cluster.
  - Resolving references to the queue manager hosting a queue inside the cluster removes the flexibility to do workload balancing.
  - It also makes it difficult for you to change a queue manager hosting a queue in the cluster.
  - The alternative is to replace `RQMNAME` with a queue manager alias provided by the cluster administrator.
  - "Hiding the name of a cluster target queue manager" describes using a queue manager alias to decouple a queue manager outside a cluster from the management of queue managers inside a cluster.
- However, the suggested way to name transmission queues is to give them the name of the target queue manager. The name of the transmission queue reveals the name of a queue manager in the cluster. You have to choose which rule to follow. You might choose to name the transmission queue using either the queue manager name or the cluster name:

  Name the transmission queue using the gateway queue manager name
  Disclosure of the gateway queue manager name to queue managers outside a cluster is a reasonable exception to the rule of hiding cluster queue manager names.

  Name the transmission queue using the name of the cluster
  If you are not following the convention of naming transmission queues with the name of the target queue manager, use the cluster name.

About this task

Modify the task "Configuring request/reply to a cluster" on page 976, to hide the name of the target queue manager inside the cluster.

Procedure

In the example, see Figure 151 on page 980, define a queue manager alias on the gateway queue manager `QM1` called `DEMO`:

```
DEFINE QREMOTE(DEMO) RNAME(' ') RQMNAME(' ')
```

The QREMOTE definition on QM1 makes the queue manager alias DEMO known to the gateway queue manager. QM3, the queue manager outside the cluster, can use the queue manager alias DEMO to send messages to cluster queues on DEMO, rather than having to use an actual queue manager name.

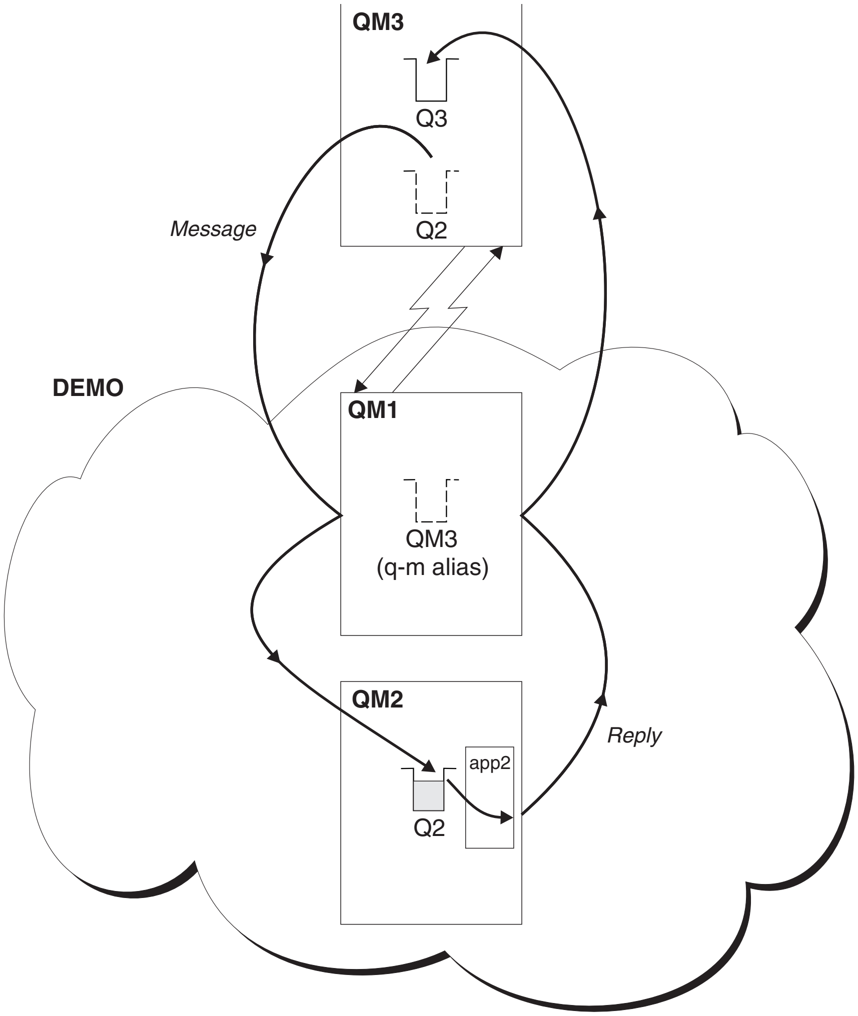


Figure 151. Putting from a queue manager outside the cluster

If you adopt the convention of using the cluster name to name the transmission queue connecting to a cluster, then the remote queue definition for Q2 becomes:

```
DEFINE QREMOTE(Q2) RNAME(Q2) RQMNAME(DEMO)
```

Results

Messages destined for Q2 on DEMO are placed on the DEMO transmission queue. From the transmission queue they are transferred by the sender-channel to the gateway queue manager, QM1. The gateway queue manager routes the messages to any queue manager in the cluster that hosts the cluster queue Q2.

Configuring request/reply from a cluster

Configure a request/reply message path from a cluster to a queue manager outside the cluster. Hide the details of how a queue manager inside the cluster communicates outside the cluster by using a gateway queue manager.

Before you begin

Figure 152 shows a queue manager, QM2, inside the cluster DEMO. It sends a request to a queue, Q3, hosted on queue manager outside the cluster. The replies are returned to Q2 at QM2 inside the cluster.

To communicate with the queue manager outside the cluster, one or more queue managers inside the cluster act as a gateway. A gateway queue manager has a communication path to the queue managers outside the cluster. In the example, QM1 is the gateway.

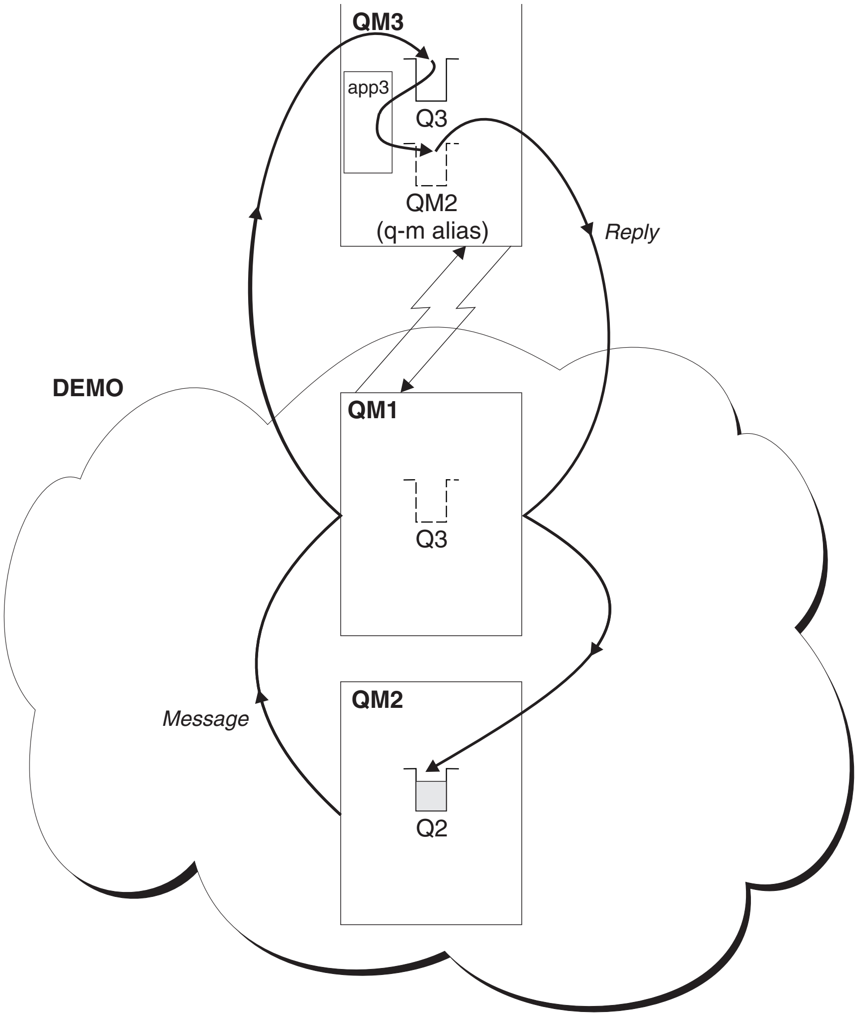


Figure 152. Putting to a queue manager outside the cluster

About this task

Follow the instructions to set up the path for the request and reply messages

Procedure

1. Send the request message from the cluster.

Consider how the queue manager, QM2, which is inside the cluster puts a message to the queue Q3 at QM3, which is outside the cluster.

a. Create a QREMOTE definition on QM1 that advertises the remote queue Q3 to the cluster

```
DEFINE QREMOTE(Q3) RNAME(Q3) RQMNAME(QM3) CLUSTER(DEMO)
```

It also has a sender-channel and a transmission queue to the queue manager that is outside the cluster. QM3 has a corresponding receiver-channel. The channels are not shown in Figure 152 on page 981.

An application on QM2 issues an MQPUT call specifying the target queue and the queue to which replies are to be sent. The target queue is Q3 and the reply-to queue is Q2.

The message is sent to QM1, which uses its remote-queue definition to resolve the queue name to Q3 at QM3.

2. Receive the reply message from the queue manager outside the cluster.

   A queue manager outside the cluster must have a queue manager alias for each queue manager in the cluster to which it send a message. The queue-manager alias must also specify the name of the transmission queue to the gateway queue manager. In this example, QM3 needs a queue manager alias definition for QM2:

   a. Create a queue manager alias QM2 on QM3

   ```
   DEFINE QREMOTE(QM2) RNAME(' ') RQMNAME(QM2) XMITQ(QM1)
   ```

   QM3 also needs a sender-channel and transmission queue to QM1 and QM1 needs a corresponding receiver-channel.

   The application, **app3**, on QM3 can then send replies to QM2, by issuing an MQPUT call and specifying the queue name, Q2 and the queue manager name, QM2.

What to do next

You can define more than one route out of a cluster.

Configuring workload balancing from outside a cluster

Configure a message path from a queue manager outside a cluster to any copy of a cluster queue. The result is to workload balance requests from outside the cluster to each instance of a cluster queue.

Before you begin

Configure the example, as shown in Figure 150 on page 977 in “Configuring request/reply to a cluster” on page 976.

About this task

In this scenario, the queue manager outside the cluster, QM3 in Figure 153 on page 983, sends requests to the queue Q2. Q2 is hosted on two queue managers within cluster DEMO to use workload balancing. A queue named Q2 is defined on the queue managers QM2 and QM4 but not on the gateway queue manager QM1. The requests from QM3, the queue manager outside the cluster, are sent to either instance of Q2.

QM3 is not part of a cluster and communicates using distributed queuing techniques. It must have a sender-channel and a transmission queue to QM1. QM1 needs a corresponding receiver-channel. The channels and transmission queues are not shown explicitly in Figure 153 on page 983.

The procedure extends the example in Figure 150 on page 977 in “Configuring request/reply to a cluster” on page 976.

Procedure

1. Define a local queue called Q2 on each of QM2 and QM4.

   ```
   DEFINE QLOCAL(Q2) CLUSTER(DEMO) DEFBIND(NOTFIXED)
   ```

2. Create a QREMOTE definition for Q2 on QM3.

   ```
   DEFINE QREMOTE(Q2) RNAME(Q2) RQMNAME(Q3) XMITQ(QM1)
   ```

   Create a QREMOTE definition for each queue in the cluster that QM3 puts messages to.
3. Create a queue-manager alias Q3 on QM3.

   ```
   DEFINE QREMOTE(Q3) RNAME(' ') RQMNAME(' ') CLUSTER(DEMO) DEFBIND(NOTFIXED)
   ```

   Q3 is not a real queue manager name. It is the name of a queue manager alias definition in the cluster that equates the queue manager alias name Q3 with blank, ' '
4. QM1, the gateway queue manager, has no special definitions.

Results

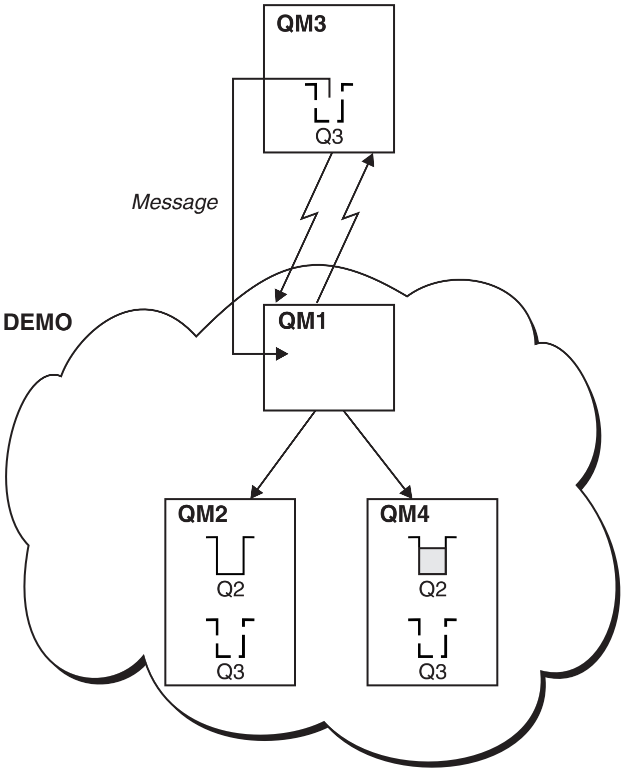


Figure 153. Putting from a queue manager outside the cluster

When an application at QM3 issues an MQPUT call to put a message to Q2, the QREMOTE definition causes the message to be routed through the gateway queue manager QM1. QM1 uses workload balancing to distribute messages targeted to Q2 between the queues called Q2 on the two queue managers, QM2 and QM4, which have cluster queue manager aliases for Q3.

Configuring message paths between clusters

Connect clusters together using a gateway queue manager. Make queues or queue managers visible to all the clusters by defining cluster queue or cluster queue manager aliases on the gateway queue manager.

About this task

Instead of grouping all your queue managers together in one large cluster, you can have many smaller clusters. Each cluster has one or more queue managers in acting as a bridge. The advantage of this is that you can restrict the visibility of queue and queue-manager names across the clusters. See "Overlapping clusters" on page 908. Use aliases to change the names of queues and queue managers to avoid name conflicts or to comply with local naming conventions.

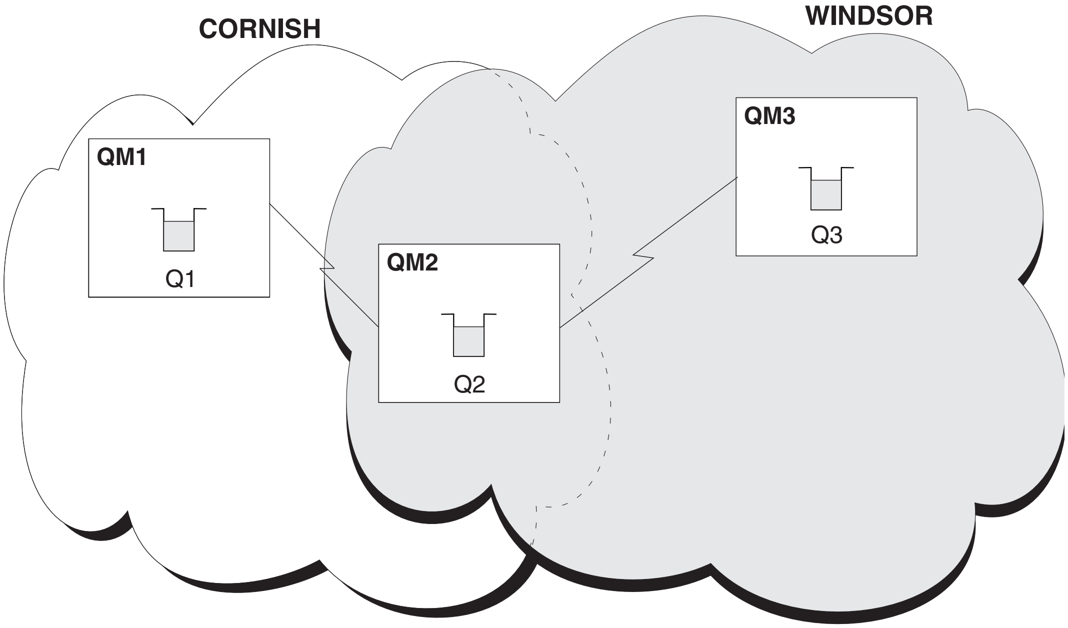


Figure 154. Bridging across clusters

Figure 154 shows two clusters with a bridge between them. There could be more than one bridge.

Configure the clusters using the following procedure:

Procedure

1. Define a cluster queue, Q1 on QM1.
   ```
   DEFINE QLOCAL(Q1) CLUSTER(CORNISH)
   ```
2. Define a cluster queue, Q3 on QM3.
   ```
   DEFINE QLOCAL(Q3) CLUSTER(WINDSOR)
   ```
3. Create a namelist called CORNISHWINDSOR on QM2, containing the names of both clusters.
   ```
   DEFINE NAMELIST(CORNISHWINDSOR) DESCR('CornishWindsor namelist')
          NAMES(CORNISH, WINDSOR)
   ```
4. Define a cluster queue, Q2 on QM2
   ```
   DEFINE QLOCAL(Q2) CLUSNL(CORNISHWINDSOR)
   ```

What to do next

QM2 is a member of both clusters and is the bridge between them. For each queue that you want to make visible across the bridge, you need a QALIAS definition on the bridge. For example in Figure 154, on QM2, you need:

```
DEFINE QALIAS(MYQ3) TARGQ(Q3) CLUSTER(CORNISH) DEFBIND(NOTFIXED)
```

Using the queue alias, an application connected to a queue manager in CORNISH, for example QM1, can put a message to Q3. It refers to Q3 as MYQ3. The message is routed to Q3 at QM3.

When you open a queue, you need to set DEFBIND to either NOTFIXED or QDEF. If DEFBIND is left as the default, OPEN, the queue manager resolves the alias definition to the bridge queue manager that hosts it. The bridge does not forward the message.

For each queue manager that you want to make visible, you need a queue-manager alias definition. For example, on QM2 you need:

```
DEFINE QREMOTE(QM1) RNAME(' ') RQMNAME(QM1) CLUSTER(WINDSOR)
```

An application connected to any queue manager in WINDSOR, for example QM3, can put a message to any queue on QM1, by naming QM1 explicitly on the MQOPEN call.

Queue-manager aliases and clusters

Use queue-manager aliases to hide the name of queue managers when sending messages into or out of a cluster, and to workload balance messages sent to a cluster.

Queue-manager aliases, which are created using a remote-queue definition with a blank RNAME, have five uses:

Remapping the queue-manager name when sending messages

A queue-manager alias can be used to remap the queue-manager name specified in an MQOPEN call to another queue manager. It can be a cluster queue manager. For example, a queue manager might have the queue-manager alias definition:

```
DEFINE QREMOTE(YORK) RNAME(' ') RQMNAME(CLUSQM)
```

YORK can be used as an alias for the queue manager called CLUSQM. When an application on the queue manager that made this definition puts a message to queue manager YORK, the local queue manager resolves the name to CLUSQM. If the local queue manager is not called CLUSQM, it puts the message on the cluster transmission queue to be moved to CLUSQM. It also changes the transmission header to say CLUSQM instead of YORK.

Note: The definition applies only on the queue manager that makes it. To advertise the alias to the whole cluster, you need to add the CLUSTER attribute to the remote-queue definition. Then messages from other queue managers that were destined for YORK are sent to CLUSQM .

Altering or specifying the transmission queue when sending messages

Aliasing can be used to join a cluster to a non-cluster system. For example, queue managers in the cluster ITALY could communicate with the queue manager called PALERMO, which is outside the cluster. To communicate, one of the queue managers in the cluster must act as a gateway. From the gateway queue manager, issue the command:

```
DEFINE QREMOTE(ROME) RNAME(' ') RQMNAME(PALERMO) XMITQ(X) CLUSTER(ITALY)
```

The command is a queue-manager alias definition. It defines and advertises ROME as a queue manager over which messages from any queue manager in the cluster ITALY can multi-hop to reach their destination at PALERMO. Messages put to a queue opened with the queue-manager name set to ROME are sent to the gateway queue manager with the queue manager alias definition. Once there, the messages are put on the transmission queue X and moved by non-cluster channels to the queue manager PALERMO .

The choice of the name ROME in this example is not significant. The values for QREMOTE and RQMNAME could both be the same.

Determining the destination when receiving messages

When a queue manager receives a message, it extracts the name of the destination queue and queue manager from the transmission header. It looks for a queue-manager alias definition with the same name as the queue manager in the transmission header. if it finds one, it substitutes the RQMNAME from the queue-manager alias definition for the queue manager name in the transmission header.

There are two reasons for using a queue-manager alias in this way:

- To direct messages to another queue manager
- To alter the queue manager name to be the same as the local queue manager

Using queue manager aliases in a gateway queue manager to route messages between queue managers in different clusters.

An application can send a message to a queue in a different cluster using a queue manager alias. The queue does not have to be a cluster queue. The queue is defined in one cluster. The application is connected to a queue manager in a different cluster. A gateway queue manager connects the two clusters. If the queue is not defined as clustered, for the correct routing to take place, the application must open the queue using the queue name and a clustered queue manager alias name. For an example of a configuration, see "Creating two-overlapping clusters with a gateway queue manager" on page 946, from which the reply message flow illustrated in figure 1, is taken.

The diagram shows the path taken by the reply message back to a temporary dynamic queue, which is called RQ. The server application, connected to QM3, opens the reply queue using the queue manager name QM2. The queue manager name QM2 is defined as a clustered queue manager alias on QM1. QM3 routes the reply message to QM1. QM1 routes the message to QM2.

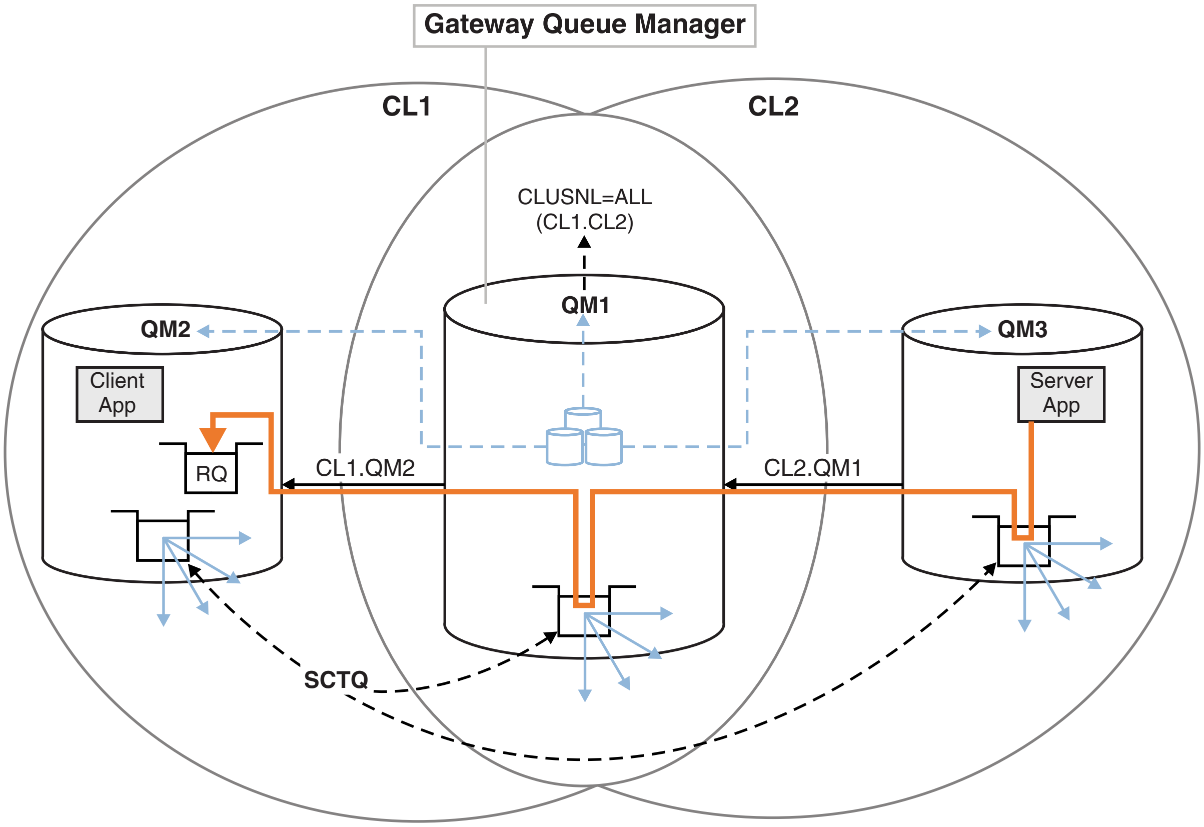


Figure 155. Using a queue manager alias to return the reply message to a different cluster

The way the routing works is as follows. Every queue manager in each cluster has a queue manager alias definition on QM1. The aliases are clustered in all the clusters. The grey dashed arrows from each of the aliases to a queue manager show that each queue manager alias is resolved to a real queue manager in at least one of the clusters. In this case, the QM2 alias is clustered in both cluster CL1 and CL2, and is resolved to the real queue manager QM2 in CL1. The server application creates the reply message using the reply to queue name RQ, and reply to queue manager name QM2. The message is routed to QM1 because the queue manager alias definition QM2 is defined on QM1 in cluster CL2 and queue manager QM2 is not in cluster CL2. As the message cannot be sent to the target queue manager, it is sent to the queue manager that has the alias definition.

QM1 places the message on the cluster transmission queue on QM1 for transferal to QM2. QM1 routes the message to QM2 because the queue manager alias definition on QM1 for QM2 defines QM2 as the real target queue manager. The definition is not circular, because alias definitions can refer only to real definitions; the alias cannot point to itself. The real definition is resolved by QM1, because both QM1 and QM2 are in the same cluster, CL1. QM1 finds out the connection information for QM2 from the repository for CL1, and routes the message to QM2. For the message to be rerouted by QM1, the server application must have opened the reply queue with the option DEFBIND set to MQBND_BIND_NOT_FIXED. If the server application had opened the reply queue with the option MQBND_BIND_ON_OPEN, the message is not rerouted and ends up on a dead letter queue.

Using a queue manager as a gateway into the cluster to workload balance messages from coming from outside the cluster.

You define a queue called EDINBURGH on more than one queue manager in the cluster. You want the clustering mechanism to balance the workload for messages coming to that queue from outside the cluster.

A queue manager from outside the cluster needs a transmit queue and sender-channel to one queue manager in the cluster. This queue is called a gateway queue manager. To take advantage of the default workload balancing mechanism, one of the following rules must apply:

- The gateway queue manager must not contain an instance of the EDINBURGH queue.
- The gateway queue manager specifies CLWLUSEQ(ANY) on ALTER QMGR.

For an example of workload balancing from outside a cluster, see “Configuring workload balancing from outside a cluster” on page 982

Reply-to queue aliases and clusters

A reply-to queue alias definition is used to specify alternative names for reply information. Reply-to queue alias definitions can be used with clusters just the same as in a distributed queuing environment.

For example:

- An application at queue manager VENICE sends a message to queue manager PISA using the MQPUT call. The application provides the following reply-to queue information in the message descriptor:

```
ReplyToQ='QUEUE'
ReplyToQMgr=''
```

- In order that replies sent to QUEUE can be received on OTHERQ at PISA, create a remote-queue definition on VENICE that is used as a reply-to queue alias. The alias is effective only on the system on which it was created.

```
DEFINE QREMOTE(QUEUE) RNAME(OTHERQ) RQMNAME(PISA)
```

RQMNAME and QREMOTE can specify the same names, even if RQMNAME is itself a cluster queue manager.

Queue aliases and clusters

Use queue aliases to hide the name of a cluster queue, to cluster a queue, adopt different attributes, or adopt different access controls.

A QALIAS definition is used to create an alias by which a queue is to be known. You might create an alias for a number of reasons:

- You want to start using a different queue but you do not want to change your applications.
- You do not want applications to know the real name of the queue to which they are putting messages.
- You might have a naming convention that differs from the one where the queue is defined.
- Your applications might not be authorized to access the queue by its real name but only by its alias.

Create a QALIAS definition on a queue manager using the DEFINE QALIAS command. For example, run the command:

```
DEFINE QALIAS(PUBLIC) TARGQ(LOCAL) CLUSTER(C)
```

The command advertises a queue called PUBLIC to the queue managers in cluster C. PUBLIC is an alias that resolves to the queue called LOCAL. Messages sent to PUBLIC are routed to the queue called LOCAL.

You can also use a queue alias definition to resolve a queue name to a cluster queue. For example, run the command:

```
DEFINE QALIAS(PRIVATE) TARGQ(PUBLIC)
```

The command enables a queue manager to use the name PRIVATE to access a queue advertised elsewhere in the cluster by the name PUBLIC. Because this definition does not include the CLUSTER attribute it applies only to the queue manager that makes it.

Using clusters for workload management

By defining multiple instances of a queue on different queue managers in a cluster you can spread the work of servicing the queue over multiple servers. There are several factors that can prevent messages being requeued to a different queue manager in the event of failure.

As well as setting up clusters to reduce system administration, you can create clusters in which more than one queue manager hosts an instance of the same queue.

You can organize your cluster such that the queue managers in it are clones of each other. Each queue manager is able to run the same applications and have local definitions of the same queues. You can spread the workload between your queue managers by having several instances of an application. Each instance of the application receives messages and runs independently of each other.

The advantages of using clusters in this way are:

- Increased availability of your queues and applications
- Faster throughput of messages
- More even distribution of workload in your network

Any one of the queue managers that hosts an instance of a particular queue can handle messages destined for that queue. Applications do not name a queue manager when sending messages. A workload management algorithm determines which queue manager handles the message.

See the following subtopics for more information about cluster configurations for workload management:

Related concepts:

"Comparison of clustering and distributed queuing" on page 888
Compare the components that need to be defined to connect queue managers using distributed queuing and clustering.

"Components of a cluster" on page 890
Clusters are composed of queue managers, cluster repositories, cluster channels, and cluster queues.

"Managing IBM WebSphere MQ clusters" on page 912
You can create, extend, and maintain IBM WebSphere MQ clusters.

"Routing messages to and from clusters" on page 975
Use queue aliases, queue manager aliases, and remote queue definitions to connect clusters to external queue managers and other clusters.

Related tasks:

"Configuring a queue manager cluster" on page 885
Use the links in this topic to find out how clusters work, how to design a cluster configuration, and to get an example of how to set up a simple cluster.

"Setting up a new cluster" on page 913
Follow these instructions to set up the example cluster. Separate instructions describe setting up the cluster on TCP/IP, LU 6.2, and with a single transmission queue or multiple transmission queues. Test the cluster works by sending a message from one queue manager to the other.

Related information:

Clusters

How clusters work

Writing and compiling cluster workload exits

Example of a cluster with more than one instance of a queue

In this example of a cluster with more than one instance of a queue, messages are routed to different instances of the queue. You can force a message to a specific instance of the queue, and you can choose to send a sequence of messages to one of either of the queue managers.

Figure 156 on page 990 shows a cluster in which there is more than one definition for the queue Q3. If an application at QM1 puts a message to Q3, it does not necessarily know which instance of Q3 is going to process its message. If an application is running on QM2 or QM4, where there are local instances of Q3, the local instance of Q3 is opened by default. By setting the CLWLUSEQ queue attribute, the local instance of the queue can be treated the same as a remote instance of the queue.

The MQOPEN option DefBind controls whether the target queue manager is chosen when the MQOPEN call is issued, or when the message is transferred from the transmission queue.

If you set DefBind to MQBND_BIND_NOT_FIXED the message can be sent to an instance of the queue that is available when the message is transmitted. This avoids the following problems:

- The target queue is unavailable when the message arrives at the target queue manager.
- The state of the queue has changed.
- The message has been put using a cluster queue alias, and no instance of the target queue exists on the queue manager where the instance of the cluster queue alias is defined.

If any if these problems are discovered at transmission time, another available instance of the target queue is sought and the message is rerouted. If no instances of the queue are available, the message is placed on the dead-letter queue.

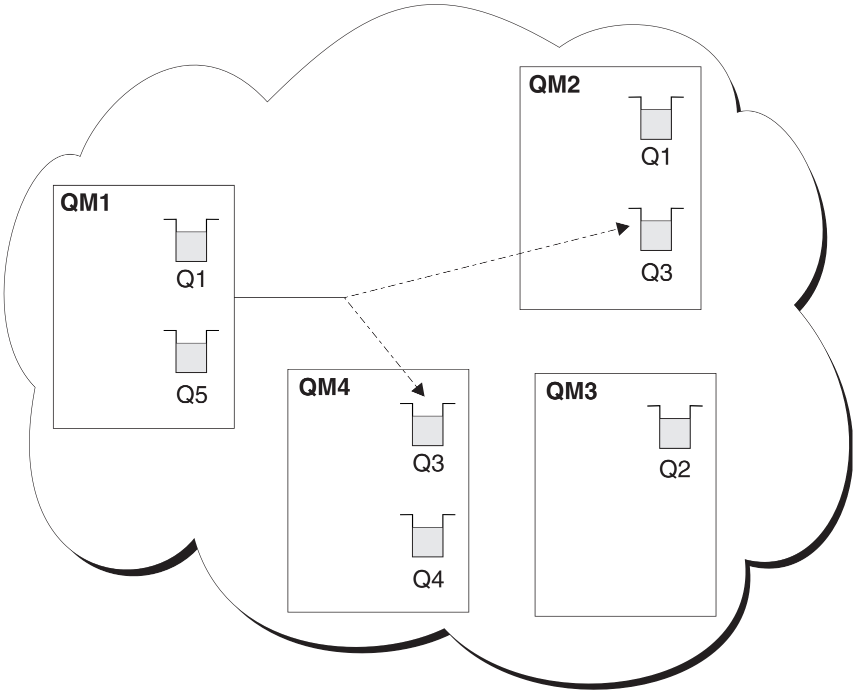


Figure 156. A cluster with multiple instances of the same queue

One factor that can prevent messages being rerouted is if messages have been assigned to a fixed queue manager or channel with `MQBND_BIND_ON_OPEN`. Messages bound on MQOPEN are never reallocated to another channel. Note also that message reallocation only takes place when a cluster channel is actually failing. Reallocation does not occur if the channel has already failed.

The system attempts to reroute a message if the destination queue manager goes out of service. In so doing, it does not affect the integrity of the message by running the risk of losing it or by creating a duplicate. If a queue manager fails and leaves a message in doubt, that message is not rerouted.

Adding a queue manager that hosts a queue locally

Follow these instructions to add an instance of `INVENTQ` to provide additional capacity to run the inventory application system in Paris and New York.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The `INVENTORY` cluster has been set up as described in Adding a new queue manager to a cluster. It contains three queue managers; `LONDON` and `NEWYORK` both hold full repositories, `PARIS` holds a partial repository. The inventory application runs on the system in New York, connected to the `NEWYORK` queue manager. The application is driven by the arrival of messages on the `INVENTQ` queue.
- We want to add an instance of `INVENTQ` to provide additional capacity to run the inventory application system in Paris and New York.

About this task

Follow these steps to add a queue manager that hosts a queue locally.

Procedure

1. Alter the PARIS queue manager.

   For the application in Paris to use the INVENTQ in Paris and the one in New York, we must inform the queue manager. On PARIS issue the following command:

   ```
   ALTER QMGR CLWLUSEQ(ANY)
   ```

2. Review the inventory application for message affinities.

   Before proceeding, ensure that the inventory application does not have any dependencies on the sequence of processing of messages. For more information, see Handling message affinities.

3. Install the inventory application on the system in Paris.
4. Define the cluster queue INVENTQ.

   The INVENTQ queue which is already hosted by the NEWYORK queue manager is also to be hosted by PARIS. Define it on the PARIS queue manager as follows:

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
   ```

   Now that you have completed all the definitions, if you have not already done so, start the channel initiator on WebSphere MQ for z/OS. On all platforms, start a listener program on queue manager PARIS. The listener listens for incoming network requests and starts the cluster-receiver channel when it is needed.

Results

Figure 157 shows the cluster set up by this task.

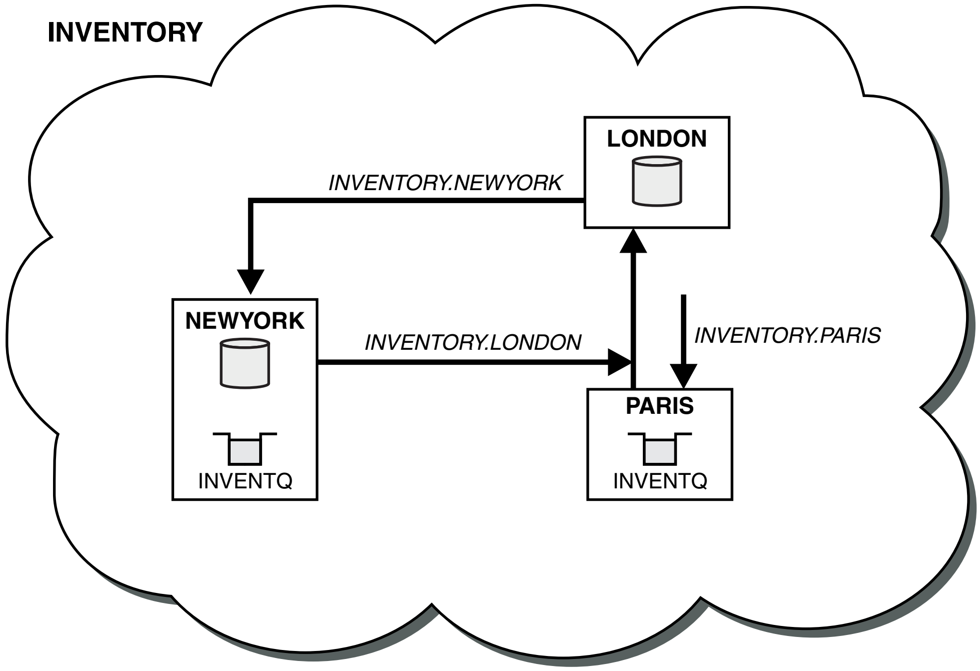


Figure 157. The INVENTORY cluster, with three queue managers

The modification to this cluster was accomplished without you altering the queue managers NEWYORK or LONDON. The full repositories in these queue managers are updated automatically with the information they need to be able to send messages to INVENTQ at PARIS.

What to do next

The INVENTQ queue and the inventory application are now hosted on two queue managers in the cluster. This increases their availability, speeds up throughput of messages, and allows the workload to be distributed between the two queue managers. Messages put to INVENTQ by any of the queue managers LONDON, NEWYORK, PARIS are routed alternately to PARIS or NEWYORK, so that the workload is balanced.

Using two networks in a cluster

Follow these instructions to add a new store in TOKYO where there are two different networks. Both need to be available for use to communicate with the queue manager in Tokyo.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster has been set up as described in "Adding a queue manager to a cluster". It contains three queue managers; LONDON and NEWYORK both hold full repositories, PARIS holds a partial repository. The inventory application runs on the system in New York, connected to the NEWYORK queue manager. The application is driven by the arrival of messages on the INVENTQ queue.
- A new store is being added in TOKYO where there are two different networks. Both need to be available for use to communicate with the queue manager in Tokyo.

About this task

Follow these steps to use two networks in a cluster.

Procedure

1. Decide which full repository TOKYO refers to first.

   Every queue manager in a cluster must refer to one or other of the full repositories to gather information about the cluster. It builds up its own partial repository. It is of no particular significance which repository you choose. In this example, NEWYORK is chosen. Once the new queue manager has joined the cluster it communicates with both of the repositories.
2. Define the CLUSRCVR channels.

   Every queue manager in a cluster needs to define a cluster-receiver on which it can receive messages. This queue manager needs to be able to communicate on each network.

   ```
   DEFINE CHANNEL(INVENTORY.TOKYO.NETB) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME('TOKYO.NETB.CMSTORE.COM') CLUSTER(INVENTORY) DESCR('Cluster-receiver channel using
   network B for TOKYO')
   ```

   ```
   DEFINE CHANNEL(INVENTORY.TOKYO.NETA) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME('TOKYO.NETA.CMSTORE.COM') CLUSTER(INVENTORY) DESCR('Cluster-receiver channel using
   network A for TOKYO')
   ```
3. Define a CLUSSDR channel on queue manager TOKYO.

   Every queue manager in a cluster needs to define one cluster-sender channel on which it can send messages to its first full repository. In this case we have chosen NEWYORK, so TOKYO needs the following definition:

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME(NEWYORK.CHSTORE.COM)
   CLUSTER(INVENTORY) DESCR('Cluster-sender channel from TOKYO to repository at NEWYORK')
   ```

   Now that you have completed all the definitions, if you have not already done so start the channel initiator on WebSphere MQ for z/OS. On all platforms, start a listener program on queue manager PARIS. The listener program listens for incoming network requests and starts the cluster-receiver channel when it is needed.

Results

Figure 158 shows the cluster set up by this task.

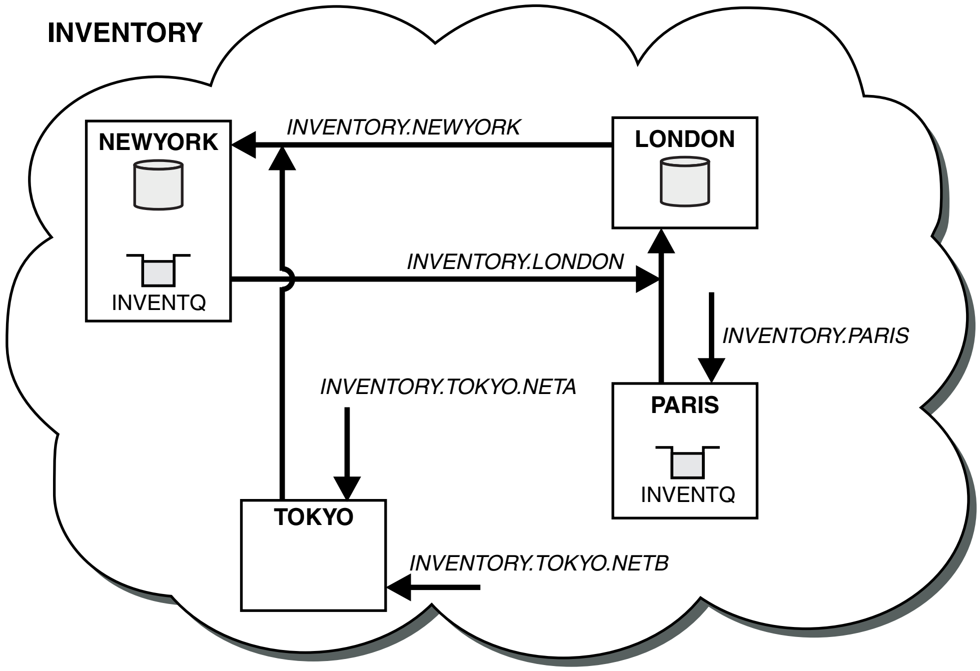


Figure 158. The `INVENTORY` cluster, with four queue managers

By making only three definitions, we have added the queue manager `TOKYO` to the cluster with two different network routes available to it.

Related tasks:

"Adding a queue manager to a cluster" on page 924
Follow these instructions to add a queue manager to the cluster you created. Messages to cluster queues and topics are transferred using the single cluster transmission queue `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

Using a primary and a secondary network in a cluster

Follow these instructions to make one network the primary network, and another network the backup network. Use the backup network if there is a problem with the primary network.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The `INVENTORY` cluster has been set up as described in "Using two networks in a cluster" on page 992. It contains four queue managers; `LONDON` and `NEWYORK` both hold full repositories; `PARIS` and `TOKYO` hold partial repositories. The inventory application runs on the system in New York, connected to the queue manager `NEWYORK`. The `TOKYO` queue manager has two different networks that it can communicate on.
- You want to make one of the networks the primary network, and another of the networks the backup network. You plan to use the backup network if there is a problem with the primary network.

About this task

Use the NETPRTY attribute to configure a primary and a secondary network in a cluster.

Procedure

Alter the existing CLUSRCVR channels on TOKYO.
To indicate that the network A channel is the primary channel, and the network B channel is the secondary channel, use the following commands:

1. `ALTER CHANNEL(INVENTORY.TOKYO.NETA) CHLTYPE(CLUSRCVR) NETPRTY(2) DESCR('Main cluster-receiver channel for TOKYO')`
2. `ALTER CHANNEL(INVENTORY.TOKYO.NETB) CHLTYPE(CLUSRCVR) NETPRTY(1) DESCR('Backup cluster-receiver channel for TOKYO')`

What to do next

By configuring the channel with different network priorities, you have now defined to the cluster that you have a primary network and a secondary network. The queue managers in the cluster that use these channels automatically use the primary network whenever it is available. The queue managers failover to use the secondary network when the primary network is not available.

Adding a queue to act as a backup

Follow these instructions to provide a backup in Chicago for the inventory system that now runs in New York. The Chicago system is only used when there is a problem with the New York system.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster has been set up as described in “Adding a queue manager to a cluster” on page 924. It contains three queue managers; LONDON and NEWYORK both hold full repositories, PARIS holds a partial repository. The inventory application runs on the system in New York, connected to the NEWYORK queue manager. The application is driven by the arrival of messages on the INVENTQ queue.
- A new store is being set up in Chicago to provide a backup for the inventory system that now runs in New York. The Chicago system only used when there is a problem with the New York system.

About this task

Follow these steps to add a queue to act as a backup.

Procedure

1. Decide which full repository CHICAGO refers to first.

   Every queue manager in a cluster must refer to one or other of the full repositories to gather information about the cluster. It builds up its own partial repository. It is of no particular significance which repository you choose for any particular queue manager. In this example, NEWYORK is chosen. Once the new queue manager has joined the cluster it communicates with both of the repositories.
2. Define the CLUSRCVR channel.

   Every queue manager in a cluster needs to define a cluster-receiver on which it can receive messages. On CHICAGO, define:

   ```
   DEFINE CHANNEL(INVENTORY.CHICAGO) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(CHICAGO.CMSTORE.COM) CLUSTER(INVENTORY) DESCR('Cluster-receiver channel for
   CHICAGO')
   ```

3. Define a CLUSSDR channel on queue manager CHICAGO.

   Every queue manager in a cluster needs to define one cluster-sender channel on which it can send messages to its first full repository. In this case we have chosen NEWYORK, so CHICAGO needs the following definition:

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME(NEWYORK.CHSTORE.COM)
   CLUSTER(INVENTORY) DESCR('Cluster-sender channel from CHICAGO to repository at NEWYORK')
   ```

4. Alter the existing cluster queue INVENTQ.

   The INVENTQ which is already hosted by the NEWYORK queue manager is the main instance of the queue.

   ```
   ALTER QLOCAL(INVENTQ) CLWLPRTY(2)
   ```

5. Review the inventory application for message affinities.

   Before proceeding, ensure that the inventory application does not have any dependencies on the sequence of processing of messages.

6. Install the inventory application on the system in CHICAGO.

7. Define the backup cluster queue INVENTQ

   The INVENTQ which is already hosted by the NEWYORK queue manager, is also to be hosted as a backup by CHICAGO. Define it on the CHICAGO queue manager as follows:

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY) CLWLPRTY(1)
   ```

   Now that you have completed all the definitions, if you have not already done so start the channel initiator on WebSphere MQ for z/OS. On all platforms, start a listener program on queue manager CHICAGO. The listener program listens for incoming network requests and starts the cluster-receiver channel when it is needed.

Results

Figure 159 shows the cluster set up by this task.

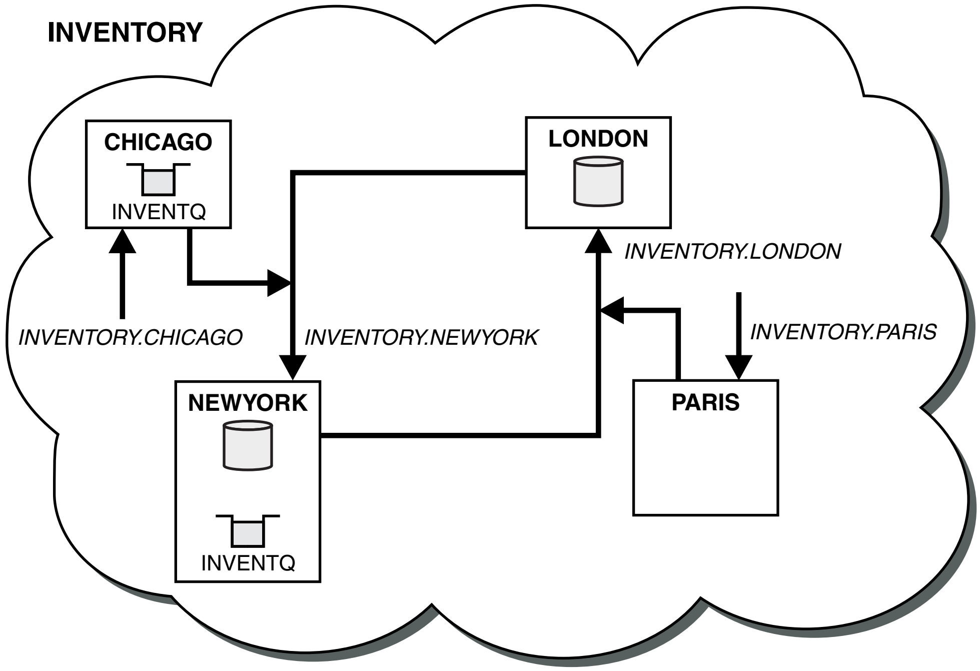


Figure 159. The INVENTORY cluster, with four queue managers

The INVENTQ queue and the inventory application are now hosted on two queue managers in the cluster. The CHICAGO queue manager is a backup. Messages put to INVENTQ are routed to NEWYORK unless it is unavailable when they are sent instead to CHICAGO.

Note:

The availability of a remote queue manager is based on the status of the channel to that queue manager. When channels start, their state changes several times, with some of the states being less preferential to the cluster workload management algorithm. In practice this means that lower priority (backup) destinations can be chosen while the channels to higher priority (primary) destinations are starting.

If you need to ensure that no messages go to a backup destination, do not use CLWLPRTY. Consider using separate queues, or CLWLRANK with a manual switch over from the primary to backup.

Restricting the number of channels used

Follow these instructions to restrict the number of active channels each server runs when a price check application is installed on various queue managers.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- A price check application is to be installed on various queue managers. To keep the number of channels being used to a low number, the number of active channels each server runs is restricted. The application is driven by the arrival of messages on the PRICEQ queue.
- Four server queue managers host the price check application. Two query queue managers send messages to the PRICEQ to query a price. Two more queue managers are configured as full repositories.

About this task

Follow these steps to restrict the number of channels used.

Procedure

1. Choose two full repositories.

   Choose two queue managers to be the full repositories for your price check cluster. They are called REPOS1 and REPOS2.

   Issue the following command:

   ```
   ALTER QMGR REPOS(PRICECHECK)
   ```
2. Define a CLUSRCVR channel on each queue manager.

   At each queue manager in the cluster, define a cluster-receiver channel and a cluster-sender channel. It does not matter which is defined first.

   ```
   DEFINE CHANNEL(PRICECHECK.SERVE1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME(SERVER1.COM)
   CLUSTER(PRICECHECK) DESCR('Cluster-receiver channel')
   ```
3. Define a CLUSSDR channel on each queue manager.

   Make a CLUSSDR definition at each queue manager to link that queue manager to one or other of the full repository queue managers.

   ```
   DEFINE CHANNEL(PRICECHECK.REPOS1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME(REPOS1.COM)
   CLUSTER(PRICECHECK) DESCR('Cluster-sender channel to repository queue manager')
   ```
4. Install the price check application.
5. Define the PRICEQ queue on all the server queue managers.

Issue the following command on each:

```
DEFINE QLOCAL(PRICEQ) CLUSTER(PRICECHECK)
```

6. Restrict the number of channels used by queries

   On the query queue managers we restrict the number of active channels used, by issuing the following commands on each:

   ```
   ALTER QMGR CLWLMRUC(2)
   ```

7. If you have not already done so, start the channel initiator on WebSphere MQ for z/OS. On all platforms, start a listener program.

   The listener program listens for incoming network requests and starts the cluster-receiver channel when it is needed.

Results

Figure 160 shows the cluster set up by this task.

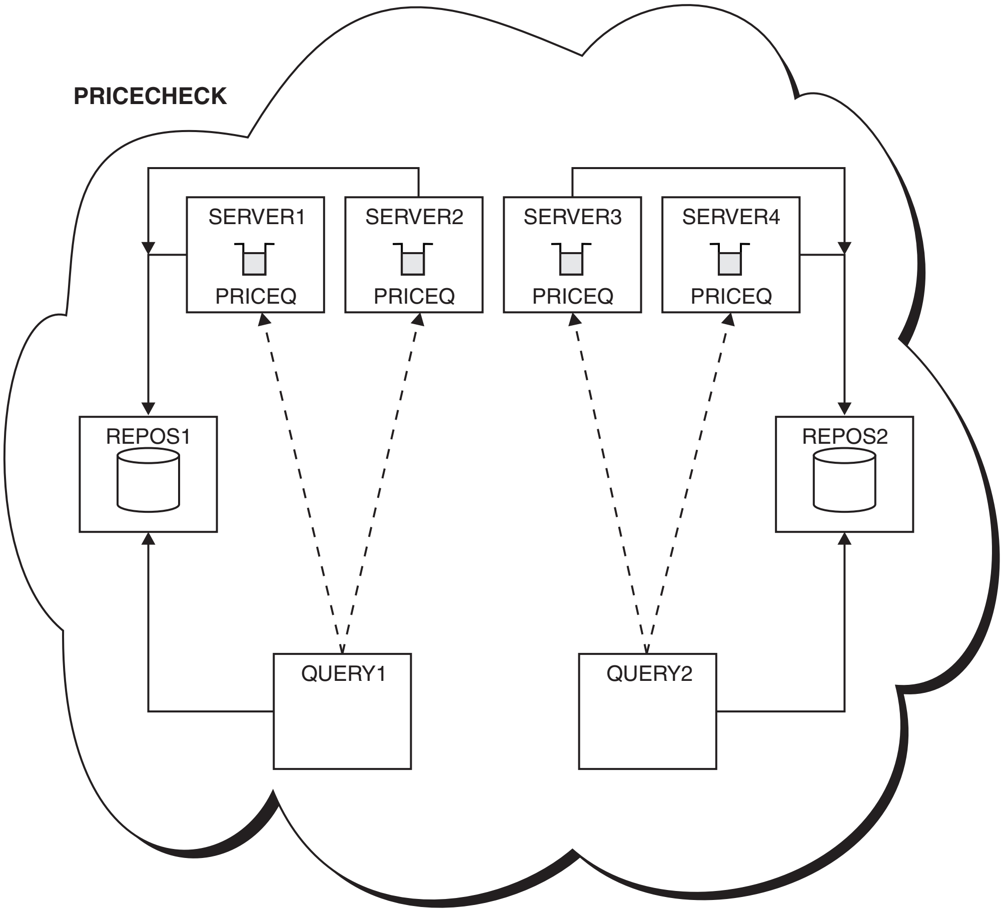


Figure 160. The `PRICECHECK` cluster, with four server queue managers, two repositories, and two query queue managers

Although there are four instances of the `PRICEQ` queue available in the `PRICECHECK` cluster, each querying queue manager only uses two of two of them. For example, the `QUERY1` queue manager only has active channels to the `SERVER1` and `SERVER2` queue managers. If `SERVER1` became unavailable, the `QUERY1` queue manager would then begin to use another queue manager, for example `SERVER3`.

What to do next

Although there are four instances of the PRICEQ queue available in the PRICECHECK cluster, each querying queue manager only uses two of two of them. For example, the QUERY1 queue manager only has active channels to the SERVER1 and SERVER2 queue managers. If SERVER1 became unavailable, the QUERY1 queue manager would then begin to use another queue manager, for example SERVER3.

Adding a more powerful queue manager that hosts a queue

Follow these instructions to provide additional capacity by running the inventory system in Los Angeles as well as New York, where Los Angeles can handle twice the number of messages as New York.

Before you begin

Note: For changes to a cluster to be propagated throughout the cluster, at least one full repository must always be available. Ensure that your repositories are available before starting this task.

Scenario:

- The INVENTORY cluster has been set up as described in "Adding a queue manager to a cluster" on page 924. It contains three queue managers: LONDON and NEWYORK both hold full repositories, PARIS holds a partial repository and puts messages from INVENTQ. The inventory application runs on the system in New York connected to the NEWYORK queue manager. The application is driven by the arrival of messages on the INVENTQ queue.
- A new store is being set up in Los Angeles. To provide additional capacity, you want to run the inventory system in Los Angeles as well as New York. The new queue manager can process twice as many messages as New York.

About this task

Follow these steps to add a more powerful queue manager that hosts a queue.

Procedure

1. Decide which full repository LOSANGELES refers to first.
2. Every queue manager in a cluster must refer to one or other of the full repositories to gather information about the cluster. It builds up its own partial repository. It is of no particular significance which repository you choose. In this example, NEWYORK is chosen. Once the new queue manager has joined the cluster it communicates with both of the repositories.

   ```
   DEFINE CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSSDR) TRPTYPE(TCP)
   CONNAME(NEWYORK.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-sender channel from LOSANGELES to repository at NEWYORK')
   ```

3. Define the CLUSRCVR channel on queue manager LOSANGELES. Every queue manager in a cluster must define a cluster-receiver channel on which it can receive messages. On LOSANGELES, define:

   ```
   DEFINE CHANNEL(INVENTORY.LOSANGELES) CHLTYPE(CLUSRCVR) TRPTYPE(TCP)
   CONNAME(LOSANGELES.CHSTORE.COM) CLUSTER(INVENTORY)
   DESCR('Cluster-receiver channel for queue manager LOSANGELES')
   CLWLWGHT(2)
   ```

   The cluster-receiver channel advertises the availability of the queue manager to receive messages from other queue managers in the cluster INVENTORY. Setting CLWLWGHT to two ensures that the Los Angeles queue manager gets twice as many of the inventory messages as New York (when the channel for NEWYORK is set to one).
4. Alter the CLUSRCVR channel on queue manager NEWYORK.

   Ensure that the Los Angeles queue manager gets twice as many of the inventory messages as New York. Alter the definition of the cluster-receiver channel.

   ```
   ALTER CHANNEL(INVENTORY.NEWYORK) CHLTYPE(CLUSRCVR) CLWLWGHT(1)
   ```

5. Review the inventory application for message affinities.

Before proceeding, ensure that the inventory application does not have any dependencies on the sequence of processing of messages.

6. Install the inventory application on the system in Los Angeles
7. Define the cluster queue INVENTQ.

   The INVENTQ queue, which is already hosted by the NEWYORK queue manager, is also to be hosted by LOSANGELES. Define it on the LOSANGELES queue manager as follows:

   ```
   DEFINE QLOCAL(INVENTQ) CLUSTER(INVENTORY)
   ```

   Now that you have completed all the definitions, if you have not already done so start the channel initiator on WebSphere MQ for z/OS. On all platforms, start a listener program on queue manager LOSANGELES. The listener program listens for incoming network requests and starts the cluster-receiver channel when it is needed.

Results

"Adding a more powerful queue manager that hosts a queue" on page 998 shows the cluster set up by this task.

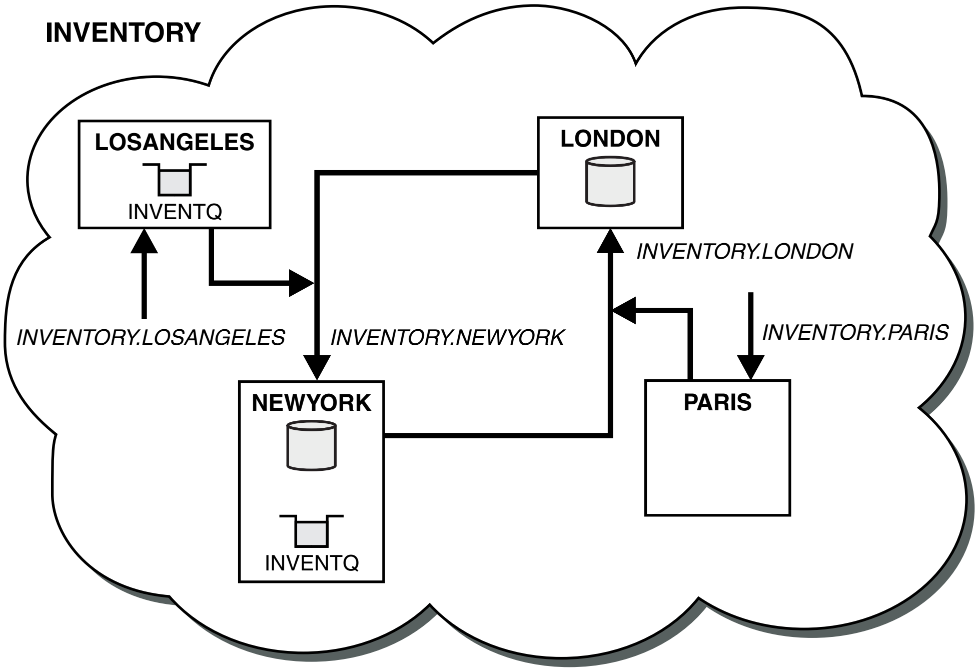


Figure 161. The INVENTORY cluster with four queue managers

This modification to the cluster was accomplished without you having to alter the queue managers LONDON and PARIS. The repositories in these queue managers are updated automatically with the information they need to be able to send messages to INVENTQ at LOSANGELES.

What to do next

The INVENTQ queue and inventory application are hosted on two queue managers in the cluster. The configuration increases their availability, speeds up throughput of messages, and allows the workload to be distributed between the two queue managers. Messages put to INVENTQ by either LOSANGELES or NEWYORK are handled by the instance on the local queue manager whenever possible. Messages put by LONDON or PARIS are routed to LOSANGELES or NEWYORK, with twice as many messages being sent to LOSANGELES.

Application programming and clusters

You do not need to make any programming changes to take advantage of multiple instances of the same queue. However, some programs do not work correctly unless a sequence of messages is sent to the same instance of a queue.

Applications can open a queue using the MQOPEN call. Applications use the MQPUT call to put messages onto an open queue. Applications can put a single message onto a queue that is not already open, using the MQPUT1 call.

If you set up clusters that have multiple instances of the same queue, there are no specific application programming considerations. However, to benefit from the workload management aspects of clustering, you might need to modify your applications. If you set up a network in which there are multiple definitions of the same queue, review your applications for message affinities.

Suppose for example, you have two applications that rely on a series of messages flowing between them in the form of questions and answers. You probably want answers to go back to the same queue manager that sent a question. It is important that the workload management routine does not send the messages to any queue manager that hosts a copy of the reply queue.

You might have applications that require messages to be processed in sequence (for example, a database replication application that sends batches of messages that must be retrieved in sequence). The use of segmented messages can also cause an affinity problem.

Opening a local or remote version of the target queue

Be aware of how the queue manager chooses whether use a local or remote version of the target queue.

1. The queue manager opens the local version of the target queue to read messages, or to set the attributes of the queue.
2. The queue manager opens any instance of the target queue to write messages to, if at least one of the following conditions is true:
   - A local version of the target queue does not exist.
   - The queue manager specifies `CLWLUSEQ(ANY)` on `ALTER QMGR`.
   - The queue on the queue manager specifies `CLWLUSEQ(ANY)`.

Handling message affinities:

Message affinities are rarely part of good programming design. You need to remove message affinities to use clustering fully. If you cannot remove message affinities, you can force related messages to be delivered using the same channel and to the same queue manager.

If you have applications with message affinities, remove the affinities before starting to use clusters.

Removing message affinities improves the availability of applications. An application sends a batch of messages that has message affinities to a queue manager. The queue manager fails after receiving only part of the batch. The sending queue manager must wait for it to recover and process the incomplete message batch before it can send any more messages.

Removing messages affinities also improves the scalability of applications. A batch of messages with affinities can lock resources at the destination queue manager while waiting for subsequent messages. These resources might remain locked for long periods of time, preventing other applications from doing their work.

Furthermore, message affinities prevent the cluster workload management routines from making the best choice of queue manager.

To remove affinities, consider the following possibilities:

- Carrying state information in the messages
- Maintaining state information in nonvolatile storage accessible to any queue manager, for example in a Db2 database
- Replicating read-only data so that it is accessible to more than one queue manager

If it is not appropriate to modify your applications to remove message affinities, there are a number of possible solutions to the problem.

Name a specific destination on the MQOPEN call

Specify the remote-queue name and the queue manager name on each MQOPEN call, and all messages put to the queue using that object handle go to the same queue manager, which might be the local queue manager.

Specifying the remote-queue name and the queue manager name on each MQOPEN call has disadvantages:

- No workload balancing is carried out. You do not take advantage of the benefits of cluster workload balancing.
- If the target queue manager is remote and there is more than one channel to it, the messages might take different routes and the sequence of messages is still not preserved.
- If your queue manager has a definition for a transmission queue with the same name as the destination queue manager, messages go on that transmission queue rather than on the cluster transmission queue.

Return the queue-manager name in the reply-to queue manager field

Allow the queue manager that receives the first message in a batch to return its name in its response. It does this using the `ReplyToQMgr` field of the message descriptor. The queue manager at the sending end can then extract the reply-to queue manager name and specify it on all subsequent messages.

Using the `ReplyToQMgr` information from the response has disadvantages:

- The requesting queue manager must wait for a response to its first message
- You must write additional code to find and use the `ReplyToQMgr` information before sending subsequent messages
- If there is more than one route to the queue manager, the sequence of the messages might not be preserved

Set the `MQOO_BIND_ON_OPEN` option on the MQOPEN call

Force all your messages to be put to the same destination using the `MQOO_BIND_ON_OPEN` option on the MQOPEN call. Either `MQOO_BIND_ON_OPEN` or `MQOO_BIND_ON_GROUP` must be specified when using message groups with clusters to ensure that all messages in the group are processed at the same destination.

By opening a queue and specifying `MQOO_BIND_ON_OPEN`, you force all messages that are sent to this queue to be sent to the same instance of the queue. `MQOO_BIND_ON_OPEN` binds all messages to the same queue manager and also to the same route. For example, if there is an IP route and a NetBIOS route to the same destination, one of these is selected when the queue is opened and this selection is honored for all messages put to the same queue using the object handle obtained.

By specifying `MQOO_BIND_ON_OPEN` you force all messages to be routed to the same destination. Therefore applications with message affinities are not disrupted. If the destination is not available, the messages remain on the transmission queue until it becomes available again.

MQOO_BIND_ON_OPEN also applies when the queue manager name is specified in the object descriptor when you open a queue. There might be more than one route to the named queue manager. For example, there might be multiple network paths or another queue manager might have defined an alias. If you specify MQOO_BIND_ON_OPEN, a route is selected when the queue is opened.

Note: This is the recommended technique. However, it does not work in a multi-hop configuration in which a queue manager advertises an alias for a cluster queue. Nor does it help in situations in which applications use different queues on the same queue manager for different groups of messages.

An alternative to specifying MQOO_BIND_ON_OPEN on the MQOPEN call, is to modify your queue definitions. On your queue definitions, specify DEFBIND(OPEN), and allow the DefBind option on the MQOPEN call to default to MQOO_BIND_AS_Q_DEF.

Set the MQOO_BIND_ON_GROUP option on the MQOPEN call

Force all your messages in a group to be put to the same destination using the MQOO_BIND_ON_GROUP option on the MQOPEN call. Either MQOO_BIND_ON_OPEN or MQOO_BIND_ON_GROUP must be specified when using message groups with clusters to ensure that all messages in the group are processed at the same destination.

By opening a queue and specifying MQOO_BIND_ON_GROUP, you force all messages in a group that are sent to this queue to be sent to the same instance of the queue. MQOO_BIND_ON_GROUP binds all messages in a group to the same queue manager, and also to the same route. For example, if there is an IP route and a NetBIOS route to the same destination, one of these is selected when the queue is opened and this selection is honored for all messages in a group put to the same queue using the object handle obtained.

By specifying MQOO_BIND_ON_GROUP you force all messages in a group to be routed to the same destination. Therefore applications with message affinities are not disrupted. If the destination is not available, the messages remain on the transmission queue until it becomes available again.

MQOO_BIND_ON_GROUP also applies when the queue manager name is specified in the object descriptor when you open a queue. There might be more than one route to the named queue manager. For example, there might be multiple network paths or another queue manager might have defined an alias. If you specify MQOO_BIND_ON_GROUP, a route is selected when the queue is opened.

If MQOO_BIND_ON_GROUP is specified but the messages are not grouped, the behaviour is equivalent to MQOO_BIND_NOT_FIXED.

Note: This is the recommended technique for ensuring that messages in a group are sent to the same destination. However, it does not work in a multi-hop configuration in which a queue manager advertises an alias for a cluster queue.

An alternative to specifying MQOO_BIND_ON_GROUP on the MQOPEN call, is to modify your queue definitions. On your queue definitions, specify DEFBIND(GROUP), and allow the DefBind option on the MQOPEN call to default to MQOO_BIND_AS_Q_DEF.

Write a customized cluster workload exit program

Instead of modifying your applications you can circumvent the message affinities problem by writing a cluster workload exit program. Writing a cluster workload exit program is not easy and is not a recommended solution. The program would have to be designed to recognize the affinity by inspecting the content of messages. Having recognized the affinity, the program would have to force the workload management utility to route all related messages to the same queue manager.

Clustering: Best practices

Clusters provide a mechanism for interconnecting queue managers. The best practices described in this section are based on testing and feedback from customers.

A successful cluster setup is dependent on good planning and a thorough understanding of IBM WebSphere MQ fundamentals, such as good application management and network design. Ensure that you are familiar with the information in the related topics listed below before continuing.

Related information:

Clustering
This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

Concepts of intercommunication

How clusters work

Clustering: Special considerations for overlapping clusters

This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

- “Cluster ownership”
- “Overlapping clusters: Gateways”
- “Cluster naming conventions” on page 1004

Cluster ownership

Familiarize yourself with overlapping clusters before reading the following information. See “Overlapping clusters” on page 908 and “Configuring message paths between clusters” on page 984 for the necessary information.

When configuring and managing a system that consists of overlapping clusters, it is best to adhere to the following:

- Although IBM WebSphere MQ clusters are 'loosely coupled' as previously described, it is useful to consider a cluster as a single unit of administration. This concept is used because the interaction between definitions on individual queue managers is critical to the smooth functioning of the cluster. For example: When using workload balanced cluster queues it is important that a single administrator or team understand the full set of possible destinations for messages, which depends on definitions spread throughout the cluster. More trivially, cluster sender/receiver channel pairs must be compatible throughout.
- Considering this previous concept; where multiple clusters meet (which are to be administered by separate teams / individuals), it is important to have clear policies in place controlling administration of the gateway queue managers.
- It is useful to treat overlapping clusters as a single namespace: Channel names and queue manager names must be unique throughout a single cluster. Administration is much easier when unique throughout the entire topology. It is best to follow a suitable naming convention, possible conventions are described in “Cluster naming conventions” on page 1004.
- Sometimes administrative and system management cooperation is essential/unavoidable: For example, cooperation between organizations that own different clusters that need to overlap. A clear understanding of who owns what, and enforceable rules/conventions helps clustering run smoothly when overlapping clusters.

Overlapping clusters: Gateways

In general, a single cluster is easier to administer than multiple clusters. Therefore creating large numbers of small clusters (one for every application for example) is something to be avoided generally.

However, to provide classes of service, you can implement overlapping clusters. For example:

- Concentric clusters where the smaller one is for Publish/Subscribe. See How to size systems: for more information.
- Some queue managers are to be administered by different teams (see "Cluster ownership" on page 1003).
- If it makes sense from an organizational or geographical point of view.
- Equivalent clusters to work with name resolution, as when implementing SSl (or TLS) in an existing cluster.

There is no security benefit from overlapping clusters; allowing clusters administered by two different teams to overlap, effectively joins the teams as well as the topology. Any:

- Name advertised in such a cluster is accessible to the other cluster.
- Name advertised in one cluster can be advertised in the other to draw off eligible messages.
- Non-advertised object on a queue manager adjacent to the gateway can be resolved from any clusters of which the gateway is a member.

The namespace is the union of both clusters and must be treated as a single namespace. Therefore, ownership of an overlapping cluster is shared amongst all the administrators of both clusters.

When a system contains multiple clusters, there might be a requirement to route messages from queue managers in one cluster to queues on queue managers in another cluster. In this situation, the multiple clusters must be interconnected in some way: A good pattern to follow is the use of gateway queue managers between clusters. This arrangement avoids building up a difficult-to-manage mesh of point-to-point channels, and provides a good place to manage such issues as security policies. There are two distinct ways of achieving this arrangement:

1. Place one (or more) queue managers in both clusters using a second cluster receiver definition. This arrangement involves fewer administrative definitions but, as previously stated, means that ownership of an overlapping cluster is shared amongst all the administrators of both clusters.
2. Pair a queue manager in cluster one with a queue manager in cluster teo using traditional point-to-point channels.

In either of these cases, various tools can be used to route traffic appropriately. In particular, queue or queue manager aliases can be used to route into the other cluster, and a queue manager alias with blank **RQMNAME** property re-drives workload balancing where it is wanted.

Cluster naming conventions

This information contains the previous guidance on naming conventions, and the current guidance. As the IBM WebSphere MQ technology improves, and as customers use technology in new or different ways, new recommendations and information must be provided for these scenarios.

Cluster naming conventions: Previous guidance

When setting up a new cluster, consider a naming convention for the queue managers. Every queue manager must have a different name, but it might help you to remember which queue managers are grouped where if you give them a set of similar names.

Every cluster-receiver channel must also have a unique name.

If you have more than one channel to the same queue manager, each with different priorities or using different protocols, you might extend the names to include the different protocols; for example `QM1.S1`, `QM1.N3`, and `QM1.T4`. In this example, `S1` might be the first SNA channel, `N3` might be the NetBIOS channel with a network priority of 3.

The final qualifier might describe the class of service the channel provides. For more information, see Defining classes of service.

Remember that all cluster-sender channels have the same name as their corresponding cluster-receiver channel.

Do not use generic connection names on your cluster-receiver definitions. In IBM WebSphere MQ for z/OS, you can define VTAM generic resources or *Dynamic Domain Name Server* (DDNS) generic names, but do not do this if you are using clusters. If you define a CLUSRCVR with a generic **CONNAME**, there is no guarantee that your CLUSSDR channels point to the queue managers that you intend. Your initial CLUSSDR might end up pointing to any queue manager in the queue-sharing group, not necessarily one that hosts a full repository. Furthermore, if a channel goes to retry status, it might reconnect to a different queue manager with the same generic name and the flow of your messages is disrupted.

Cluster naming conventions: Current guidance

The previous guidance in the section, "Cluster naming conventions: Previous guidance" on page 1004, is still valid. However the following guidance is intended as an update when designing new clusters. This updated suggestion ensures uniqueness of channels across multiple clusters, allowing multiple clusters to be successfully overlapped. Because queue managers and clusters can have names of up to 48 characters, and a channel name is limited to 20 characters, care must be taken when naming objects from the beginning to avoid having to change the naming convention midway through a project.

When setting up a new cluster, consider a naming convention for the queue managers. Every queue manager must have a different name. If you give queue managers in a cluster a set of similar names, it might help you to remember which queue managers are grouped where.

When defining channels, remember that all automatically created cluster-sender channels on any queue manager in the cluster have the same name as their corresponding cluster-receiver channel configured on the receiving queue manager in the cluster, and must therefore be unique and make sense across the cluster to the administrators of that cluster. Channel names are limited to a maximum of 20 characters.

One possibility is to use the queue-manager name preceded by the cluster-name. For example, if the cluster-name is CLUSTER1 and the queue-managers are QM1, QM2, then cluster-receiver channels are CLUSTER1.QM1, CLUSTER1.QM2.

You might extend this convention if channels have different priorities or use different protocols; for example, `CLUSTER1.QM1.S1`, `CLUSTER1.QM1.N3`, and `CLUSTER1.QM1.T4`. In this example, `S1` might be the first SNA channel, `N3` might be the NetBIOS channel with a network priority of three.

A final qualifier might describe the class of service the channel provides.

Clustering: Topology design considerations

This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

By thinking about where user applications and internal administrative processes are going to be located in advance, many problems can either be avoided, or minimized at a later date. This topic contains information about design decisions that can improve performance, and simplify maintenance tasks as the cluster scales.

- "Performance of the clustering infrastructure" on page 1006
- "Full repositories" on page 1006
- "Should applications use queues on full repositories?" on page 1007
- "Managing channel definitions" on page 1008
- "Workload balancing over multiple channels" on page 1008

Performance of the clustering infrastructure

When an application tries to open a queue on a queue manager in a cluster, the queue manager registers its interest with the full repositories for that queue so that it can learn where the queue exists in the cluster. Any updates to the queue location or configuration are automatically sent by the full repositories to the interested queue manager. This registering of interest is internally known as a subscription (these subscriptions are not the same as IBM WebSphere MQ subscriptions used for publish/subscribe messaging in IBM WebSphere MQ)

All information about a cluster goes through every full repository. Full repositories are therefore always being used in a cluster for administrative message traffic. The high usage of system resources when managing these subscriptions, and the transmission of them and the resulting configuration messages, can cause a considerable load on the clustering infrastructure. There are a number of things to consider when ensuring that this load is understood and minimized wherever possible:

- The more individual queue managers using a cluster queue, the more subscriptions are in the system, and thus the bigger the administrative overhead when changes occur and interested subscribers need to be notified, especially on the full repository queue managers. One way to minimize unnecessary traffic and full repository load is by connecting similar applications (that is, those applications that work with the same queues) to a smaller number of queue managers.
- In addition to the number of subscriptions in the system affecting the performance the rate of change in the configuration of clustered objects can affect performance, for example the frequent changing of a clustered queue configuration.
- When a queue manager is a member of multiple clusters (that is, it is part of an overlapping cluster system) any interest made in a queue results in a subscription for each cluster it is a member of, even if the same queue managers are the full repositories for more than one of the clusters. This arrangement increases the load on the system, and is one reason to consider whether multiple overlapping clusters are necessary, rather than a single cluster.
- Application message traffic (that is, the messages being sent byIBM WebSphere MQ applications to the cluster queues) does not go via the full repositories to reach the destination queue managers. This message traffic is sent directly between the queue manager where the message enters the cluster, and the queue manager where the cluster queue exists. It is not therefore necessary to accommodate high rates of application message traffic with respect to the full repository queue managers, unless the full repository queue managers happen to be either of those two queue managers mentioned. For that reason, it is recommended that full repository queue managers are not used for application message traffic in clusters where the clustering infrastructure load is significant.

Full repositories

A repository is a collection of information about the queue managers that are members of a cluster. A queue manager that hosts a complete set of information about every queue manager in the cluster has a full repository. For more information about full repositories and partial repositories, see "Cluster repository" on page 890.

Full repositories must be held on servers that are reliable and as highly available as possible and single points of failure must be avoided. The cluster design must always have two full repositories. If there is a failure of a full repository, the cluster can still operate.

Details of any updates to cluster resources made by a queue manager in a cluster; for example, clustered queues, are sent from that queue manager to two full repositories at most in that cluster (or to one if there is only one full repository queue manager in the cluster). Those full repositories hold the information and propagate it to any queue managers in the cluster that show an interest in it (that is, they subscribe to it). To ensure that each member of the cluster has an up-to-date view of the cluster resources there, each queue manager must be able to communicate with at least one full repository queue manager at any one time.

If, for any reason a queue manager cannot communicate with any full repositories, it can continue to function in the cluster based on its already cached level of information for a period time, but no new updates or access to previously unused cluster resources are available.

For this reason, you must aim to keep the two full repositories available at all times. However, this arrangement does not mean that extreme measures must be taken because the cluster functions adequately for a short while without a full repository.

There is another reason that a cluster must have two full repository queue managers, other than the availability of cluster information: This reason is to ensure that the cluster information held in the full repository cache exists in two places for recovery purposes. If there is only one full repository, and it loses its information about the cluster, then manual intervention on all queue managers within the cluster is required in order to get the cluster working again. If there are two full repositories however, then because information is always published to and subscribed for from two full repositories, the failed full repository can be recovered with the minimum of effort.

- It is possible to perform maintenance on full repository queue managers in a two full repository cluster design without impacting users of that cluster: The cluster continues to function with only one repository, so where possible bring the repositories down, apply the maintenance, and back up again one at a time. Even if there is an outage on the second full repository, running applications are unaffected for a minimum of three days.
- Unless there is a good reason for using a third repository, such as using a geographically local full repository for geographical reasons, use the two repository design. Having three full repositories means that you never know which are the two that are currently in use, and there might be administrative problems caused by interactions between multiple workload management parameters. It is not recommend to have more than two full repositories.
- If you still need better availability, consider hosting the full repository queue managers as multi-instance queue managers or using platform specific high availability support to improve their availability.
- You must fully interconnect all the full repository queue managers with manually defined cluster sender channels. Particular care must be taken when the cluster does have, for some justifiable reason, more than two full repositories. In this situation it is often possible to miss one or more channels and for it not to be immediately apparent. When full interconnection does not occur, hard to diagnose problems often arise. They are hard to diagnose because some full repositories not holding all repository data and therefore resulting in queue managers in the cluster having different views of the cluster depending on the full repositories that they connect to.

Should applications use queues on full repositories?

A full repository is in most ways exactly like any other queue manager, and it is therefore possible to host application queues on the full repository and connect applications directly to these queue managers. Should applications use queues on full repositories?

The commonly accepted answer is "No?. Although this configuration is possible, many customers prefer to keep these queue managers dedicated to maintaining the full repository cluster cache. Points to consider when deciding on either option are described here, but ultimately the cluster architecture must be appropriate to the particular demands of the environment.

- Upgrades: Usually, in order to use new cluster features in new releases of IBM WebSphere MQ the full repository queue managers of that cluster must be upgraded first. When an application in the cluster wants to use new features, it might be useful to be able to update the full repositories (and some subset of partial repositories) without testing a number of co-located applications.
- Maintenance: In a similar way if you must apply urgent maintenance to the full repositories, they can be restarted or refreshed with the **REFRESH** command without touching applications.

- Performance: As clusters grow and demands on the full repository cluster cache maintenance become greater, keeping applications separate reduces risk of this affecting application performance through contention for system resources.
- Hardware requirements: Typically, full repositories do not need to be powerful; for example, a simple UNIX server with a good expectation of availability is sufficient. Alternatively, for very large or constantly changing clusters, the performance of the full repository computer must be considered.
- Software requirements: Requirements are usually the main reason for choosing to host application queues on a full repository. In a small cluster, collocation might mean a requirement for fewer queue managers/servers over all.

Managing channel definitions

Even within a single cluster, multiple channel definitions can exist giving multiple routes between two queue managers.

There is sometimes an advantage to having parallel channels within a single cluster, but this design decision must be considered thoroughly; apart from adding complexity, this design might result in channels being under-utilized which reduces performance. This situation occurs because testing usually involves sending lots of messages at a constant rate, so the parallel channels are fully used. But with real-world conditions of a non-constant stream of messages, the workload balancing algorithm causes performance to drop as the message flow is switched from channel to channel.

When a queue manager is a member of multiple clusters, the option exists to use a single channel definition with a cluster namelist, rather than defining a separate CLUSRCVR channel for each cluster. However, this setup can cause administration difficulties later; consider for example the case where SSL is to be applied to one cluster but not a second. It is therefore preferable to create separate definitions, and the naming convention suggested in “Cluster naming conventions” on page 1004 supports this.

Workload balancing over multiple channels

This information is intended as an advanced understanding of the subject. For the basic explanation of this subject (which must be understood before using the information here), see “Using clusters for workload management” on page 988, Workload balancing , and The cluster workload management algorithm.

The cluster workload management algorithm provides a large set of tools, but they must not all be used with each other without fully understanding how they work and interact. It might not be immediately obvious how important channels are to the workload balancing process: The workload management round-robin algorithm behaves as though multiple cluster channels to a queue manager that owns a clustered queue, are treated as multiple instances of that queue. This process is explained in more detail in the following example:

1. There are two queue managers hosting a queue in a cluster: `QM1` and `QM2`.
2. There are five cluster receiver channels to `QM1`.
3. There is only one cluster receiver channel to `QM2`.
4. When **`MQPUT`** or **`MQOPEN`** on `QM3` chooses an instance, the algorithm is five times more likely to send the message to `QM1` than to `QM2`.
5. The situation in step 4 occurs because the algorithm sees six options to choose from (5+1) and round-robins across all five channels to `QM1` and the single channel to `QM2`.

Another subtle behavior is that even when putting messages to a clustered queue that happens to have one instance configured on the local queue manager, IBM WebSphere MQ uses the state of the local cluster receiver channel to decide whether messages are to be put to the local instance of the queue or remote instances of the queue. In this scenario:

1. When putting messages the workload management algorithm does not look at individual cluster queues, it looks at the cluster channels which can reach those destinations.
2. To reach local destinations, the local receiver channels are included in this list (although they are not used to send the message).
3. When a local receiver channel is stopped, the workload management algorithm, prefers an alternative instance by default if its CLUSRCVR is not stopped. If there are multiple local CLUSRCVR instances for the destination and at least one is not stopped, the local instance remains eligible.

Clustering: Application isolation using multiple cluster transmission queues

You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

When you deploy an application, you have a choice over which IBM WebSphere MQ resources it shares with other applications and which resources it does not share. There are a number of types of resources that can be shared, the main ones being the server itself, the queue manager, channels, and queues. You can choose to configure applications with fewer shared resources; allocating separate queues, channels, queue managers, or even servers to individual applications. If you do so, the overall system configuration becomes bigger and more complex. Using IBM WebSphere MQ clusters reduces the complexity of managing more servers, queue managers, queues, and channels, but it introduces another shared resource, the cluster transmission queue, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

Figure 162 on page 1010 is a slice out of a large IBM WebSphere MQ deployment that illustrates the significance of sharing `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In the diagram, the application, `Client App`, is connected to the queue manager `QM2` in cluster `CL1`. A message from `Client App` is processed by the application, `Server App`. The message is retrieved by `Server App` from the cluster queue `Q1` on the queue manager `QM3` in `CLUSTER2`. Because the client and server applications are not in the same cluster, the message is transferred by the gateway queue manager `QM1`.

The normal way to configure a cluster gateway is to make the gateway queue manager a member of all the clusters. On the gateway queue manager are defined clustered alias queues for cluster queues in all the clusters. The clustered queue aliases are available in all the clusters. Messages put to the cluster queue aliases are routed via the gateway queue manager to their correct destination. The gateway queue manager puts messages sent to the clustered alias queues onto the common `SYSTEM.CLUSTER.TRANSMIT.QUEUE` on `QM1`.

The hub and spoke architecture requires all messages between clusters to pass through the gateway queue manager. The result is that all messages flow through the single cluster transmission queue on `QM1`, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

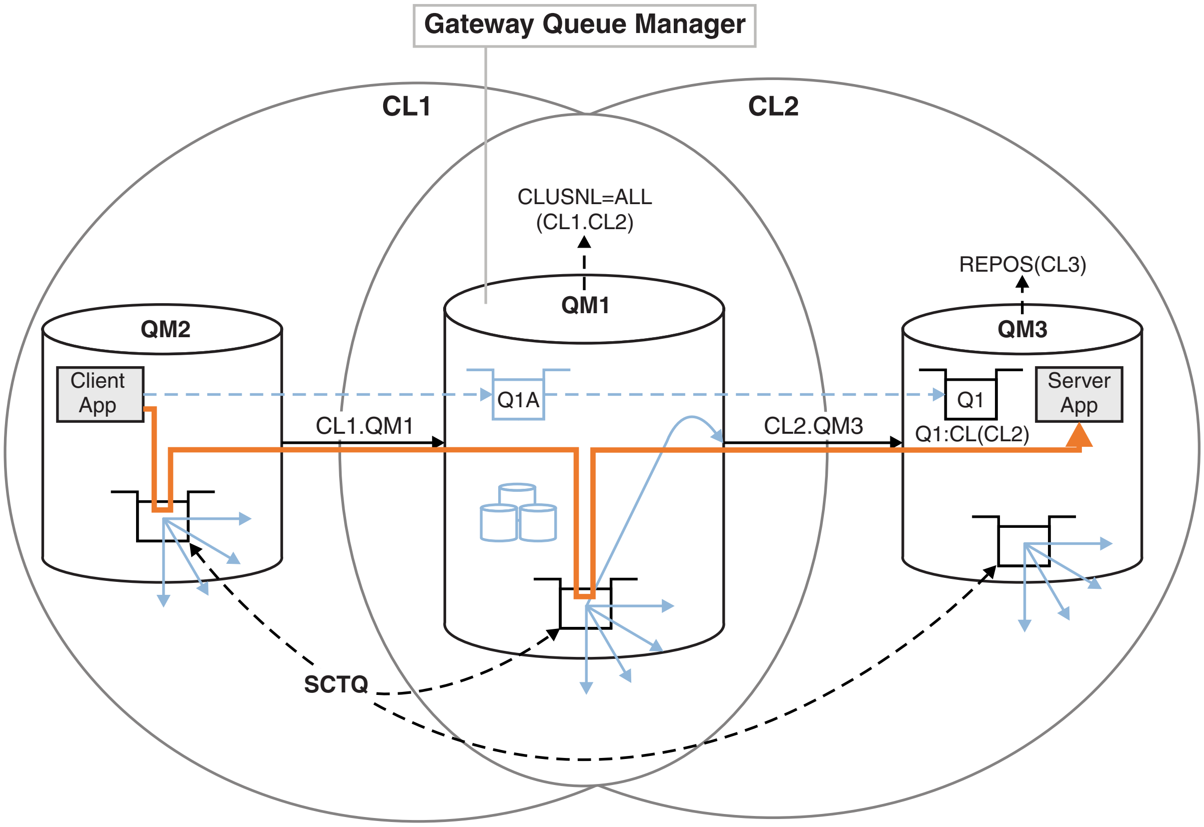
From a performance perspective, a single queue is not a problem. A common transmission queue generally does not represent a performance bottleneck. Message throughput on the gateway is largely determined by the performance of the channels that connect to it. Throughput is not generally affected by the number of queues, or the number of messages on the queues that use the channels.

From some other perspectives, using a single transmission queue for multiple applications has drawbacks:

- You cannot isolate the flow of messages to one destination from the flow of messages to another destination. You cannot separate the storage of messages before they are forwarded, even if the destinations are in different clusters on different queue managers.

  If one cluster destination becomes unavailable, messages for that destination build-up in the single transmission queue, and eventually the messages fill it up. Once the transmission queue is full, it stops messages from being placed onto the transmission queue for any cluster destination.

- It is not easy to monitor the transfer of messages to different cluster destinations. All the messages are on the single transmission queue. Displaying the depth of the transmission queue gives you little indication whether messages are being transferred to all destinations.



Note: The arrows in Figure 162 and following figures are of different types. Solid arrows represent message flows. The labels on solid arrows are message channel names. The gray solid arrows are potential message flows from the SYSTEM.CLUSTER.TRANSMIT.QUEUE onto cluster-sender channels. Black dashed lines connect labels to their targets. Gray dashed arrows are references; for example from an MQOPEN call by Client App to the cluster alias queue definition Q1A.

Figure 162. Client-server application deployed to hub and spoke architecture using IBM WebSphere MQ clusters

In Figure 162, clients of Server App open the queue Q1A. Messages are put to SYSTEM.CLUSTER.TRANSMIT.QUEUE on QM2, transferred to SYSTEM.CLUSTER.TRANSMIT.QUEUE on QM1, and then transferred to Q1 on QM3, where they are received by the Server App application.

The message from Client App passes through system cluster transmission queues on QM2 and QM1. In Figure 162, the objective is to isolate the message flow on the gateway queue manager from the client application, so that its messages are not stored on SYSTEM.CLUSTER.TRANSMIT.QUEUE. You can isolate flows on any of the other clustered queue managers. You can also isolate flows in the other direction, back to the client. To keep the descriptions of the solutions brief, the descriptions consider only a single flow from the client application.

Solutions to isolating cluster message traffic on a cluster gateway queue manager

One way to solve the problem is to use queue manager aliases, or remote queue definitions, to bridge between clusters. Create a clustered remote queue definition, a transmission queue, and a channel, to separate each message flow on the gateway queue manager; see “Adding a remote queue definition to isolate messages sent from a gateway queue manager” on page 928.

From Version 7.5 onwards, cluster queue managers are not limited to a single cluster transmission queue. You have two choices:

1. Define additional cluster transmission queues manually, and define which cluster-sender channels transfer messages from each transmission queue; see “Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager” on page 931.
2. Allow the queue manager to create and manage additional cluster transmission queues automatically. It defines a different cluster transmission queue for each cluster-sender channel; see “Changing the default to separate cluster transmission queues to isolate message traffic” on page 951

You can combine manually defined cluster transmission queues for some cluster-sender channels, with the queue manager managing the rest. The combination of transmission queues is the approach taken in “Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager” on page 931. In that solution, most messages between clusters use the common `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. One application is critical, and all its message flows are isolated from other flows by using one manually defined cluster transmission queue.

The configuration in “Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager” on page 931is limited. It does not separate the message traffic going to a cluster queue on the same queue manager in the same cluster as another cluster queue. You can separate the message traffic to individual queues by using the remote queue definitions that are part of distributed queueing. With clusters, using multiple cluster transmission queues, you can separate message traffic that goes to different cluster-sender channels. Multiple cluster queues in the same cluster, on the same queue manager, share a cluster-sender channel. Messages for those queues are stored on the same transmission queue, before being forwarded from the gateway queue manager. In the configuration in “Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager” on page 934, the limitation is side-stepped by adding another cluster and making the queue manager and cluster queue a member of the new cluster. The new queue manager might be the only queue manager in the cluster. You could add more queue managers to the cluster, and use the same cluster to isolate cluster queues on those queue managers as well.

Related concepts:

“Access control and multiple cluster transmission queues” on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, or checking against local profiles for the cluster queue, or cluster queue manager.

“Clustering: Special considerations for overlapping clusters” on page 1003
This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

“Cluster transmission queues and cluster-sender channels” on page 898
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

“Overlapping clusters” on page 908
Overlapping clusters provide additional administrative capabilities. Use namelists to reduce the number of commands needed to administer overlapping clusters.

Related tasks:

“Adding a remote queue definition to isolate messages sent from a gateway queue manager” on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

"Changing the default to separate cluster transmission queues to isolate message traffic" on page 951
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Configuring message paths between clusters" on page 984
Connect clusters together using a gateway queue manager. Make queues or queue managers visible to all the clusters by defining cluster queue or cluster queue manager aliases on the gateway queue manager.

"Clustering: Planning how to configure cluster transmission queues"
You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Related reference:

"Security" on page 1157
Use the `Security` stanza in the `qm.ini` file to specify options for the Object Authority Manager (OAM).

Related information:

Authorizing putting messages on remote cluster queues

setmqaut

Clustering: Planning how to configure cluster transmission queues

You are guided through the choices of cluster transmission queues. You can configure one common default queue, separate default queues, or manually defined queues. Configuring multiple cluster transmission queues applies to platforms other than z/OS.

Before you begin

Review "How to choose what type of cluster transmission queue to use" on page 1015.

About this task

You have some choices to make when you are planning how to configure a queue manager to select a cluster transmission queue.

1. What is the default cluster transmission queue for cluster message transfers?
   a. A common cluster transmission queue, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.
   b. Separate cluster transmission queues. The queue manager manages the separate cluster transmission queues. It creates them as permanent-dynamic queues from the model queue, `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`. It creates one cluster transmission queue for each cluster-sender channel it uses.

2. For the cluster transmission queues that you do decide to create manually, you have a further two choices:
   a. Define a separate transmission queue for each cluster-sender channel that you decide to configure manually. In this case, set the **CLCHNAME** queue attribute of the transmission queue to the name of a cluster-sender channel. Select the cluster-sender channel that is to transfer messages from this transmission queue.
   b. Combine message traffic for a group of cluster-sender channels onto the same cluster transmission queue; see Figure 163. In this case, set the **CLCHNAME** queue attribute of each common transmission queue to a generic cluster-sender channel name. A generic cluster-sender channel name is a filter to group cluster-sender channel names. For example, `SALES.*` groups all cluster-sender channels that have names beginning with `SALES.`. You can place multiple wildcard characters anywhere in the filter-string. The wildcard character is an asterisk, "*". It represents from zero to any number of characters.

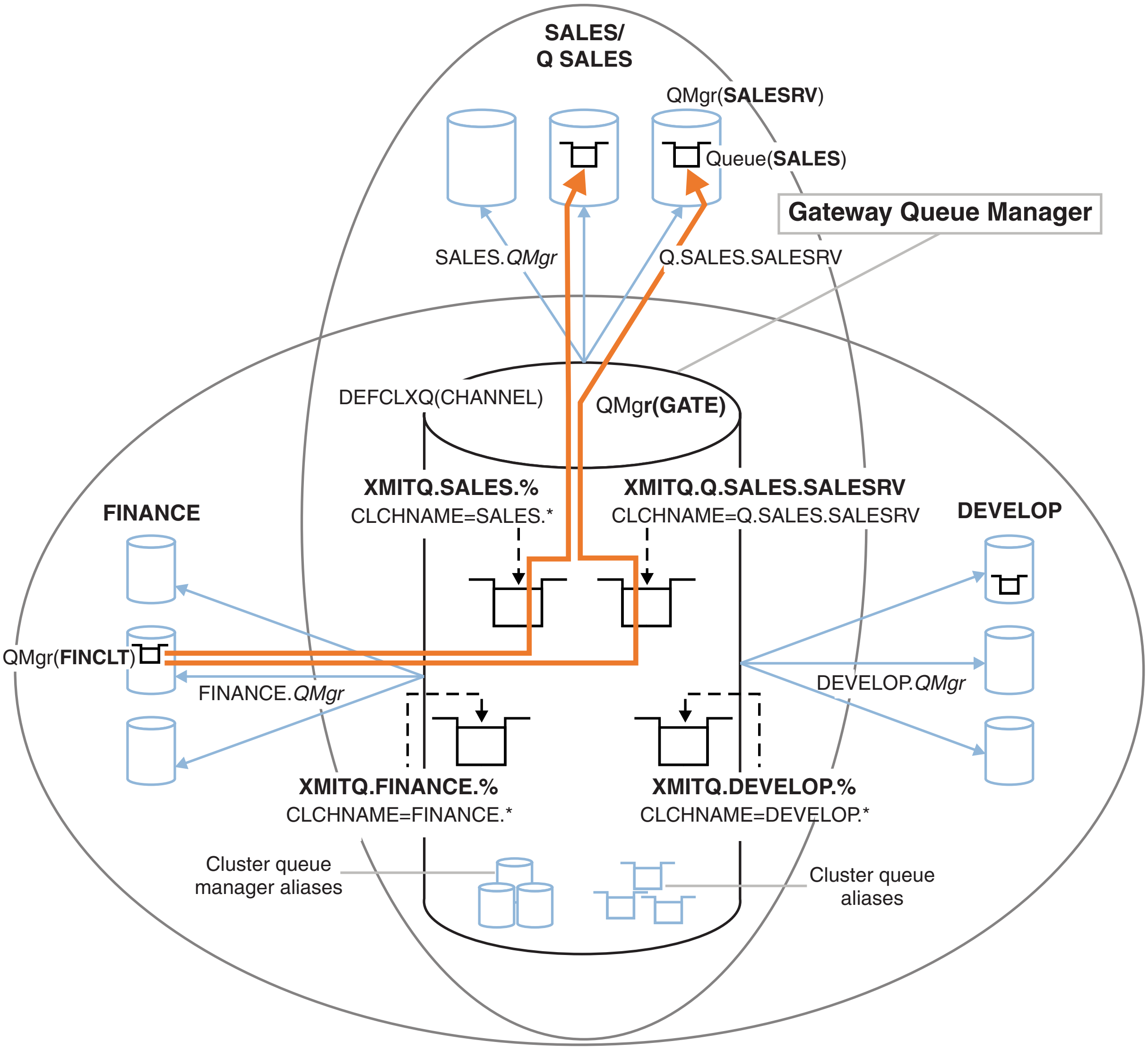


Figure 163. Example of specific transmission queues for different departmental IBM WebSphere MQ clusters

Procedure

1. Select the type of default cluster transmission queue to use.
   - Choose a single cluster transmission queue, or separate queues for each cluster connection.

   Leave the default setting or run the **MQSC** command:

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```
2. Isolate any message flows that must not share a cluster transmission queue with other flows.
   - See "Clustering: Example configuration of multiple cluster transmission queues" on page 1017. In the example the SALES queue, which must be isolated, is a member of the SALES cluster, on SALESRV. To isolate the SALES queue, create a new cluster Q.SALES, make the SALESRV queue manager a member, and modify the SALES queue to belong to Q.SALES.
   - Queue managers that send messages to SALES must also be members of the new cluster. If you use a clustered queue alias and a gateway queue manager, as in the example, in many cases you can limit the changes to making the gateway queue manager a member of the new cluster.
   - However, separating flows from the gateway to the destination does not separate flows to the gateway from the source queue manager. But it sometimes turns out to be sufficient to separate flows from the gateway and not flows to the gateway. If it is not sufficient, then add the source queue manager into the new cluster. If you want messages to travel through the gateway, move the cluster alias to the new cluster and continue to send messages to the cluster alias on the gateway, and not directly to the target queue manager.

   Follow these steps to isolate message flows:

   a. Configure the destinations of the flows so that each target queue is the only queue in a specific cluster, on that queue manager.

   b. Create the cluster-sender and cluster-receiver channels for any new clusters you have created following a systematic naming convention.
      - See "Clustering: Special considerations for overlapping clusters" on page 1003.

   c. Define a cluster transmission queue for each isolated destination on every queue manager that sends messages to the target queue.
      - A naming convention for cluster transmission queues is to use the value of the cluster channel name attribute, CLCHNAME, prefixed with XMITQ.
3. Create cluster transmission queues to meet governance or monitoring requirements.
   - Typical governance and monitoring requirements result in a transmission queue per cluster or a transmission queue per queue manager. If you follow the naming convention for cluster channels, *ClusterName.QueueManagerName*, it is easy to create generic channel names that select a cluster of queue managers, or all the clusters a queue manager is a member of; see "Clustering: Example configuration of multiple cluster transmission queues" on page 1017.
   - Extend the naming convention for cluster transmission queues to cater for generic channel names, by replacing the asterisk symbol by a percent sign. For example,

     ```
     DEFINE QLOCAL(XMITQ.SALES.%) USAGE(XMITQ) CLCHNAME(SALES.*)
     ```

Related concepts:

"Cluster transmission queues and cluster-sender channels" on page 898
Messages between clustered queue managers are stored on cluster transmission queues and forwarded by cluster-sender channels.

"Clustering: Application isolation using multiple cluster transmission queues" on page 1009
You can isolate the message flows between queue managers in a cluster. You can place messages being transported by different cluster-sender channels onto different cluster transmission queues. You can use the approach in a single cluster or with overlapping clusters. The topic provides examples and some best practices to guide you in choosing an approach to use.

"Access control and multiple cluster transmission queues" on page 887
Choose between three modes of checking when an application puts messages to remote cluster queues. The modes are checking remotely against the cluster queue, checking locally against `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, or checking against local profiles for the cluster queue, or cluster queue manager.

"Clustering: Special considerations for overlapping clusters" on page 1003
This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

"Overlapping clusters" on page 908
Overlapping clusters provide additional administrative capabilities. Use namelists to reduce the number of commands needed to administer overlapping clusters.

Related tasks:

"Adding a remote queue definition to isolate messages sent from a gateway queue manager" on page 928
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses a clustered queue remote definition, and a separate sender channel and transmission queue.

"Adding a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 931
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster transmission queue to separate message traffic to a single queue manager in a cluster.

"Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934
Modify the configuration of overlapping clusters that use a gateway queue manager. After the modification messages are transferred to an application from the gateway queue manager without using the same transmission queue or channels as other cluster messages. The solution uses an additional cluster to isolate the messages to a particular cluster queue.

"Changing the default to separate cluster transmission queues to isolate message traffic" on page 951
You can change the default way a queue manager stores messages for a clustered queue or topic on a transmission queue. Changing the default provides you with a way to isolate cluster messages on a gateway queue manager.

"Creating two-overlapping clusters with a gateway queue manager" on page 946
Follow the instructions in the task to construct overlapping clusters with a gateway queue manager. Use the clusters as a starting point for the following examples of isolating messages to one application from messages to other applications in a cluster.

"Configuring message paths between clusters" on page 984
Connect clusters together using a gateway queue manager. Make queues or queue managers visible to all the clusters by defining cluster queue or cluster queue manager aliases on the gateway queue manager.

How to choose what type of cluster transmission queue to use:

How to choose between different cluster transmission queue configuration options.

From Version 7.5 onwards, you can choose which cluster transmission queue is associated with a cluster-sender channel.

1. You can have all cluster-sender channels associated with the single default cluster transmit queue, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. This option is the default, and is the only choice for queue managers that run version Version 7.1, or earlier.
2. You can set all cluster-sender channels to be automatically associated with a separate cluster transmission queue. The queues are created by the queue manager from the model queue

SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE and named SYSTEM.CLUSTER.TRANSMIT.*ChannelName*. Channels will use their uniquely named cluster transmit queue if the queue manager attribute **DEFCLXQ** is set to CHANNEL.
3. You can set specific cluster-sender channels to be served by a single cluster transmission queue. Select this option by creating a transmission queue and setting its **CLCHNAME** attribute to the name of the cluster-sender channel.
4. You can select groups of cluster-sender channels to be served by a single cluster transmission queue. Select this option by creating a transmission queue and setting its **CLCHNAME** attribute to a generic channel name, such as *ClusterName*.*. If you name cluster channels by following the naming conventions in "Clustering: Special considerations for overlapping clusters" on page 1003, this name selects all cluster channels connected to queue managers in the cluster *ClusterName*.

You can combine either of the default cluster transmission queue options for some cluster-sender channels, with any number of specific and generic cluster transmission queue configurations.

Best practices

In most cases, for existing IBM WebSphere MQ installations, the default configuration is the best choice. A cluster queue manager stores cluster messages on a single cluster transmission queue, SYSTEM.CLUSTER.TRANSMIT.QUEUE. You have the choice of changing the default to storing messages for different queue managers and different clusters on separate transmission queues, or of defining your own transmission queues.

In most cases, for new IBM WebSphere MQ installations, the default configuration is also the best choice. The process of switching from the default configuration to the alternative default of having one transmission queue for each cluster-sender channel is automatic. Switching back is also automatic. The choice of one or the other is not critical, you can reverse it.

The reason for choosing a different configuration is more to do with governance, and management, than with functionality or performance. With a couple of exceptions, configuring multiple cluster transmission queues does not benefit the behavior of the queue manager. It results in more queues, and requires you to modify monitoring and management procedures you have already set up that refer to the single transmission queue. That is why, on balance, remaining with the default configuration is the best choice, unless you have strong governance or management reasons for a different choice.

The exceptions are both concerned with what happens if the number of messages stored on SYSTEM.CLUSTER.TRANSMIT.QUEUE increases. If you take every step to separate the messages for one destination from the messages for another destination, then channel and delivery problems with one destination ought not to affect the delivery to another destination. However, the number of messages stored on SYSTEM.CLUSTER.TRANSMIT.QUEUE can increase due to not delivering messages fast enough to one destination. The number of messages on SYSTEM.CLUSTER.TRANSMIT.QUEUE for one destination can affect the delivery of messages to other destinations.

To avoid problems that result from filling up a single transmission queue, aim to build sufficient capacity into your configuration. Then, if a destination fails and a message backlog starts to build up, you have time to fix the problem.

If messages are routed through a hub queue manager, such as a cluster gateway, they share a common transmission queue, SYSTEM.CLUSTER.TRANSMIT.QUEUE. If the number of messages stored on SYSTEM.CLUSTER.TRANSMIT.QUEUE on the gateway queue manager reaches its maximum depth, the queue manager starts to reject new messages for the transmission queue until its depth reduces. The congestion affects messages for all destinations that are routed through the gateway. Messages back up the transmission queues of other queue managers that are sending messages to the gateway. The problem manifests itself in messages written to queue manager error logs, falling message throughput, and longer elapsed times between sending a message and the time that a message arrives at its destination.

The effect of congestion on a single transmission queue can become apparent, even before it is full. If you have a mixed message traffic, with some large non-persistent messages and some small messages, the time to deliver small messages increases as the transmission queue fills. The delay is due to writing large non-persistent messages to disk that would not normally get written to disk. If you have time critical message flows, sharing a cluster transmission queue with other mixed messages flows, it could be worth configuring a special message path to isolate it from other message flows; see "Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934

The other reasons for configuring separate cluster transmission queues are to meet governance requirements, or to simplify monitoring messages that are sent to different cluster destinations. For example, you might have to demonstrate that messages for one destination never share a transmission queue with messages for another destination.

Change the queue manager attribute **DEFCLXQ** that controls the default cluster transmission queue, to create different cluster transmission queues for every cluster-sender channel. Multiple destinations can share a cluster-sender channel, so you must plan your clusters to meet this objective fully. Apply the method "Adding a cluster and a cluster transmit queue to isolate cluster message traffic sent from a gateway queue manager" on page 934 systematically to all your cluster queues. The result you are aiming for is for no cluster destination to share a cluster-sender channel with another cluster destination. As a consequence, no message for a cluster destination shares its cluster transmission queue with a message for another destination.

Creating a separate cluster transmission queue for some specific message flow, makes it easy to monitor the flow of messages to that destination. To use a new cluster transmission queue, define the queue, associate it with a cluster-sender channel, and stop and start the channel. The change does not have to be permanent. You could isolate a message flow for a while, to monitor the transmission queue, and then revert to using the default transmission queue again.

Clustering: Example configuration of multiple cluster transmission queues:

In this task you apply the steps to plan multiple cluster transmission queues to three overlapping clusters. The requirements are to separate messages flows to one cluster queue, from all other message flows, and to store messages for different clusters on different cluster transmission queues.

About this task

The steps in this task show how to apply the procedure in "Clustering: Planning how to configure cluster transmission queues" on page 1012 and arrive at the configuration shown in Figure 164 on page 1018. It is an example of three overlapping clusters, with a gateway queue manager, that is configured with separate cluster transmission queues. The MQSC commands to define the clusters are described in "Creating the example clusters" on page 1020.

For the example, there are two requirements. One is to separate the message flow from the gateway queue manager to the sales application that logs sales. The second is to query how many messages are waiting to be sent to different departmental areas at any point in time. The `SALES`, `FINANCE`, and `DEVELOP` clusters are already defined. Cluster messages are currently forwarded from `SYSTEM.CLUSTER.TRANSMIT.QUEUE`.

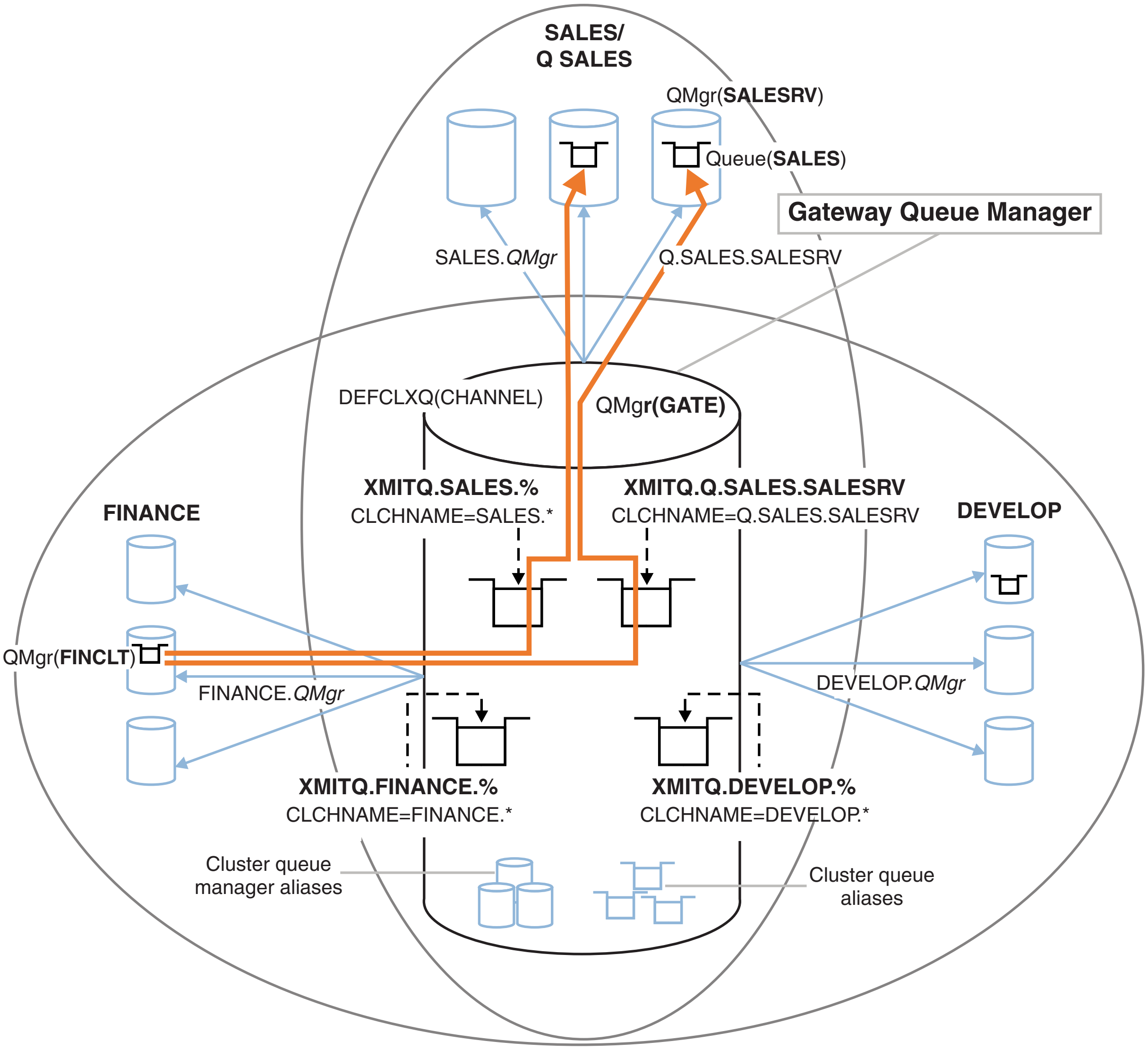


Figure 164. Example of specific transmission queues for different departmental IBM WebSphere MQ clusters

The steps to modify the clusters are as follows; see Figure 166 on page 1022 for the definitions.

Procedure

1. The first configuration step is to “ Select the type of default cluster transmission queue to use”.

   The decision is to create separate default cluster transmission queues by running the following **MQSC** command on the GATE queue manager.

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

   There is no strong reason for choosing this default, as the intention is to manually define cluster transmission queues. The choice does have a weak diagnostic value. If a manual definition is done wrongly, and a message flows down a default cluster transmission queue, it shows up in the creation of a permanent-dynamic cluster transmission queue.
2. The second configuration step is to “ Isolate any message flows that must not share a cluster transmission queue with other flows”. In this case the sales application that receives messages from the queue SALES on SALESRV requires isolation. Only isolation of messages from the gateway queue manager is required. The three sub-steps are:

a. “Configure the destinations of the flows so that each target queue is the only queue in a specific cluster, on that queue manager”.

   The example requires adding the queue manager SALESRV to a new cluster within the sales department. If you have few queues that require isolation, you might decide on creating a specific cluster for the SALES queue. A possible naming convention for the cluster name is to name such clusters, Q. *QueueName*, for example Q.SALES . An alternative approach, which might be more practical if you have a large number of queues to be isolated, is to create clusters of isolated queues where and when needed. The clusters names might be QUEUES.*n*.

   In the example, the new cluster is called Q.SALES. To add the new cluster, see the definitions in Figure 166 on page 1022. The summary of definition changes is as follows:

   1) Add Q.SALES to the namelist of clusters on the repository queue managers. The namelist is referred to in the queue manager **REPOSNL** parameter.
   2) Add Q.SALES to the namelist of clusters on the gateway queue manager. The namelist is referred to in all the cluster queue alias and cluster queue manager alias definitions on the gateway queue manager.
   3) Create a namelist on the queue manager SALESRV , for both the clusters it is a member of, and change the cluster membership of the SALES queue:

      ```
      DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES) REPLACE
      ALTER QLOCAL(SALES) CLUSTER(' ') CLUSNL(SALESRV.CLUSTERS)
      ```

      The SALES queue is a member of both clusters, just for the transition. Once the new configuration is running, you remove the SALES queue from the SALES cluster; see Figure 167 on page 1022.

b. “Create the cluster-sender and cluster-receiver channels for any new clusters you have created following a systematic naming convention”.

   1) Add the cluster-receiver channel Q.SALES.*RepositoryQMgr* to each of the repository queue managers
   2) Add the cluster-sender channel Q.SALES.*OtherRepositoryQMgr* to each of the repository queue managers, to connect to the other repository manager. Start these channels.
   3) Add the cluster receiver channels Q.SALES.SALESRV , and Q.SALES.GATE to either of the repository queue managers that is running.
   4) Add the cluster-sender channels Q.SALES.SALESRV , and Q.SALES.GATE to the SALESRV and GATE queue managers. Connect the cluster-sender channel to the repository queue manager that you created the cluster-receiver channels on.

c. “Define a cluster transmission queue for each isolated destination on every queue manager that sends messages to the target queue”.

   On the gateway queue manager define the cluster transmission queue XMITQ.Q.SALES.SALESRV for the Q.SALES.SALESRV cluster-sender channel:

   ```
   DEFINE QLOCAL(XMITQ.Q.SALES.SALESRV) USAGE(XMITQ) CLCHNAME(Q.SALES.SALESRV) REPLACE
   ```

3. The third configuration step is to “ Create cluster transmission queues to meet governance or monitoring requirements”.

   On the gateway queue manager define the cluster transmission queues:

   ```
   DEFINE QLOCAL(XMITQ.SALES)   USAGE(XMITQ) CLCHNAME(SALES.*)   REPLACE
   DEFINE QLOCAL(XMITQ.DEVELOP) USAGE(XMITQ) CLCHNAME(DEVELOP.*) REPLACE
   DEFINE QLOCAL(XMITQ.FINANCE) USAGE(XMITQ) FINANCE(SALES.*)    REPLACE
   ```

What to do next

Switch to the new configuration on the gateway queue manager.

The switch is triggered by starting the new channels, and restarting the channels that are now associated with different transmission queues. Alternatively, you can stop and start the gateway queue manager.

1. Stop the following channels on the gateway queue manager:

   ```
   SALES.Qmgr
   DEVELOP.Qmgr
   FINANCE.Qmgr
   ```

2. Start the following channels on the gateway queue manager:

   ```
   SALES.Qmgr
   DEVELOP.Qmgr
   FINANCE.Qmgr
   Q.SALES.SAVESRV
   ```

When the switch is complete, remove the SALES queue from the SALES cluster; see Figure 167 on page 1022.

Creating the example clusters:

The definitions and instructions to create the example cluster, and modify it to isolate the SALES queue and separate messages on the gateway queue manager.

About this task

The full **MQSC** commands to create the FINANCE, SALES, and Q.SALES clusters are provided in Figure 165 on page 1021, Figure 166 on page 1022, and Figure 167 on page 1022. The definitions for the basic clusters are in Figure 165 on page 1021. The definitions in Figure 166 on page 1022 modify the basic clusters to isolate the SALES queue, and to separate cluster messages to FINANCE and SALES. Finally, to remove the SALES queue from the SALES cluster; see Figure 167 on page 1022. The DEVELOP cluster is omitted from the definitions, to keep the definitions shorter.

Procedure

1. Create the SALES and FINANCE clusters, and the gateway queue manager.
   a. Create the queue managers.

      Run the command: `crtmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE` *QmgrName* for each of the queue manager names in Table 128.

Table 128. Queue manager names and port numbers

Description	Queue Manager Name	Port number
Finance repository	FINR1	1414
Finance repository	FINR2	1415
Finance client	FINCLT	1418
Sales repository	SALER1	1416
Sales repository	SALER2	1417
Sales server	SALESRV	1419
Gateway	GATE	1420

   b. Start all the queue managers

      Run the command: `strmqm` *QmgrName* for each of the queue manager names in Table 128.

   c. Create the definitions for each of the queue managers

      Run the command: `runmqsc` *QmgrName* < *filename* where the files are listed in Figure 165 on page 1021, and the file name matches the queue manager name.

finr1.txt

```
DEFINE LISTENER(1414) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1414) REPLACE
START LISTENER(1414)
ALTER QMGR REPOS(FINANCE)
DEFINE CHANNEL(FINANCE.FINR2) CHLTYPE(CLUSSDR)  CONNAME('localhost(1415)') CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1414)') CLUSTER(FINANCE) REPLACE
```

finr2.txt

```
DEFINE LISTENER(1415) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1415) REPLACE
START LISTENER(1415)
ALTER QMGR REPOS(FINANCE)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1414)') CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINR2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1415)') CLUSTER(FINANCE) REPLACE
```

finclt.txt

```
DEFINE LISTENER(1418) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1418) REPLACE
START LISTENER(1418)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR)   CONNAME('localhost(1414)') CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINCLT) CHLTYPE(CLUSRCVR) CONNAME('localhost(1418)') CLUSTER(FINANCE) REPLACE
DEFINE QMODEL(SYSTEM.SAMPLE.REPLY) REPLACE
```

saler1.txt

```
DEFINE LISTENER(1416) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1416) REPLACE
START LISTENER(1416)
ALTER QMGR REPOS(SALES)
DEFINE CHANNEL(SALES.SALER2) CHLTYPE(CLUSSDR)  CONNAME('localhost(1417)') CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1416)') CLUSTER(SALES) REPLACE
```

saler2.txt

```
DEFINE LISTENER(1417) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1417) REPLACE
START LISTENER(1417)
ALTER QMGR REPOS(SALES)
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1416)') CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALER2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1417)') CLUSTER(SALES) REPLACE
```

salesrv.txt

```
DEFINE LISTENER(1419) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1419) REPLACE
START LISTENER(1419)
DEFINE CHANNEL(SALES.SALER1)  CHLTYPE(CLUSSDR)  CONNAME('localhost(1416)') CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALESRV) CHLTYPE(CLUSRCVR) CONNAME('localhost(1419)') CLUSTER(SALES) REPLACE
DEFINE QLOCAL(SALES) CLUSTER(SALES) TRIGGER INITQ(SYSTEM.DEFAULT.INITIATION.QUEUE) PROCESS(ECHO) REPLACE
DEFINE PROCESS(ECHO) APPLICID(AMQSECH) REPLACE
```

gate.txt

```
DEFINE LISTENER(1420) TRPTYPE(TCP) IPADDR(LOCALHOST) CONTROL(QMGR) PORT(1420) REPLACE
START LISTENER(1420)
DEFINE NAMELIST(ALL) NAMES(SALES, FINANCE)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR)  CONNAME('LOCALHOST(1414)') CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.GATE)  CHLTYPE(CLUSRCVR) CONNAME('LOCALHOST(1420)') CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(SALES.SALER1)  CHLTYPE(CLUSSDR)  CONNAME('LOCALHOST(1416)') CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.GATE)    CHLTYPE(CLUSRCVR) CONNAME('LOCALHOST(1420)') CLUSTER(SALES) REPLACE
DEFINE QALIAS(A.SALES) CLUSNL(ALL) TARGET(SALES) TARGTYPE(QUEUE) DEFBIND(NOTFIXED) REPLACE
DEFINE QREMOTE(FINCLT)  RNAME(' ') RQMNAME(FINCLT)  CLUSNL(ALL) REPLACE
DEFINE QREMOTE(SALESRV) RNAME(' ') RQMNAME(SALESRV) CLUSNL(ALL) REPLACE
```

Figure 165. Definitions for the basic clusters

2. Test the configuration by running the sample request program.
   a. Start the trigger monitor program on the `SALESRV` queue manager

      On Windows, open a command window and run the command `runmqtrm -m SALESRV`
   b. Run the sample request program, and send a request.

      On Windows, open a command window and run the command `amqsreq A.SALES FINCLT`

      The request message is echoed back, and after 15 seconds the sample program finishes.

3. Create the definitions to isolate the SALES queue in the Q.SALES cluster and separate cluster messages for the SALES and FINANCE cluster on the gateway queue manager.

   Run the command: `runmqsc` *QmgrName* < *filename* where the files are listed in Figure 166, and the file name almost matches the queue manager name.

chgsaler1.txt

```
DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES)
ALTER  QMGR REPOS(' ') REPOSNL(CLUSTERS)
DEFINE CHANNEL(Q.SALES.SALER2) CHLTYPE(CLUSSDR)  CONNAME('localhost(1417)') CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1416)') CLUSTER(Q.SALES) REPLACE
```

chgsaler2.txt

```
DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES)
ALTER  QMGR REPOS(' ') REPOSNL(CLUSTERS)
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1416)') CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SALER2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1417)') CLUSTER(Q.SALES) REPLACE
```

chgsalesrv.txt

```
DEFINE NAMELIST (CLUSTERS) NAMES(SALES, Q.SALES)
DEFINE CHANNEL(Q.SALES.SALER1)  CHLTYPE(CLUSSDR)  CONNAME('localhost(1416)') CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SAVESRV) CHLTYPE(CLUSRCVR) CONNAME('localhost(1419)') CLUSTER(Q.SALES) REPLACE
ALTER  QLOCAL (SALES) CLUSTER(' ') CLUSNL(CLUSTERS)
```

chggate.txt

```
ALTER  NAMELIST(ALL) NAMES(SALES, FINANCE, Q.SALES)
ALTER  QMGR DEFCLXQ(CHANNEL)
DEFINE CHANNEL(Q.SALES.SALER1)  CHLTYPE(CLUSSDR)   CONNAME('localhost(1416)') CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.GATE)    CHLTYPE(CLUSRCVR)  CONNAME('localhost(1420)') CLUSTER(Q.SALES) REPLACE
DEFINE QLOCAL (XMITQ.Q.SALES.SALESRV) USAGE(XMITQ) CLCHNAME(Q.SALES.SALESRV) REPLACE
DEFINE QLOCAL (XMITQ.SALES)         USAGE(XMITQ) CLCHNAME(SALES.*)   REPLACE
DEFINE QLOCAL (XMITQ.FINANCE)       USAGE(XMITQ) CLCHNAME(FINANCE.*) REPLACE
```

Figure 166. Changes to isolate the sales queue in a new cluster and separate the gateway cluster transmission queues

4. Remove the SALES queue from the SALES cluster.

   Run the **MQSC** command in Figure 167:

```
ALTER QLOCAL(SALES) CLUSTER('Q.SALES') CLUSNL(' ')
```

Figure 167. Remove the sales queue on queue manager SALESRV from the sales cluster

5. Switch the channels to the new transmission queues.

   The requirement is to stop and start all the channels that the GATE queue manager is using. To do this with the least number of commands, stop and start the queue manager

   ```
   endmqm -i GATE
   strmqm GATE
   ```

What to do next

1. Rerun the sample request program to verify the new configuration works; see step 2 on page 1021
2. Monitor the messages flowing through all the cluster transmission queues on the GATE queue manager:
   a. Alter the definition of each of the cluster transmission queues to turn queue monitoring on.

      ```
      ALTER QLOCAL(SYSTEM.CLUSTER.TRANSMIT.name) STATQ(ON)
      ```

   b. Check queue manager statistics monitoring is OFF , to minimize output, and set the monitoring interval to a lower value to perform multiple tests conveniently.

      ```
      ALTER QMGR STATINT(60) STATCHL(OFF) STATQ(OFF) STATMQI(OFF) STATACLS(OFF)
      ```

   c. Restart the GATE queue manager.

d. Run the sample request program a few times to verify that an equal number of messages are flowing through SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SALESRV and SYSTEM.CLUSTER.TRANSMIT.QUEUE. Requests flow through SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SALESRV and replies through SYSTEM.CLUSTER.TRANSMIT.QUEUE.

```
amqsmon -m GATE -t statistics
```

e. The results over a couple of intervals are as follows:

```
C:\Documents and Settings\Admin>amqsmon -m GATE -t statistics
MonitoringType: QueueStatistics
QueueManager: 'GATE'
IntervalStartDate: '2012-02-27'
IntervalStartTime: '14.59.20'
IntervalEndDate: '2012-02-27'
IntervalEndTime: '15.00.20'
CommandLevel: 700
ObjectCount: 2
QueueStatistics: 0
  QueueName: 'SYSTEM.CLUSTER.TRANSMIT.QUEUE'
  CreateDate: '2012-02-24'
  CreateTime: '15.58.15'
  ...
  Put1Count: [0, 0]
  Put1FailCount: 0
  PutBytes: [435, 0]
  GetCount: [1, 0]
  GetBytes: [435, 0]
  ...
QueueStatistics: 1
  QueueName: 'SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV'
  CreateDate: '2012-02-24'
  CreateTime: '16.37.43'
  ...
  PutCount: [1, 0]
  PutFailCount: 0
  Put1Count: [0, 0]
  Put1FailCount: 0
  PutBytes: [435, 0]
  GetCount: [1, 0]
  GetBytes: [435, 0]
  ...
MonitoringType: QueueStatistics
QueueManager: 'GATE'
IntervalStartDate: '2012-02-27'
IntervalStartTime: '15.00.20'
IntervalEndDate: '2012-02-27'
IntervalEndTime: '15.01.20'
CommandLevel: 700
ObjectCount: 2
QueueStatistics: 0
  QueueName: 'SYSTEM.CLUSTER.TRANSMIT.QUEUE'
  CreateDate: '2012-02-24'
  CreateTime: '15.58.15'
  ...
  PutCount: [2, 0]
  PutFailCount: 0
  Put1Count: [0, 0]
```

```
    Put1FailCount: 0
    PutBytes: [863, 0]
    GetCount: [2, 0]
    GetBytes: [863, 0]
    ...
  QueueStatistics: 1
    QueueName: 'SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV'
    CreateDate: '2012-02-24'
    CreateTime: '16.37.43'
    ...
    PutCount: [2, 0]
    PutFailCount: 0
    Put1Count: [0, 0]
    Put1FailCount: 0
    PutBytes: [863, 0]
    GetCount: [2, 0]
    GetBytes: [863, 0]
    ...
  2 Records Processed.
```

One request and reply message were sent in the first interval and two in the second. You can infer that the request messages were placed on SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV , and the reply messages on SYSTEM.CLUSTER.TRANSMIT.QUEUE .

Clustering: Switching cluster transmission queues:

Plan how the changes to the cluster transmission queues of an existing production queue manager are going to be brought into effect.

Before you begin

If you reduce the number of messages the switching process has to transfer to the new transmission queue, switching completes more quickly. Read “How the process to switch cluster-sender channel to a different transmission queue works” on page 900 for the reasons for trying to empty the transmission queue before proceeding any further.

About this task

You have a choice of two ways of making the changes to cluster transmission queues take effect.

1. Let the queue manager make the changes automatically. This is the default. The queue manager switches cluster-sender channels with pending transmission queue changes when a cluster-sender channel next starts.
2. Make the changes manually. You can make the changes to a cluster-sender channel when it is stopped. You can switch it from one cluster transmission queue to another before the cluster-sender channel starts.

What factors do you take into account when deciding which of the two options to choose, and how do you manage the switch?

Procedure

- Option 1: Let the queue manager make the changes automatically; see “Switching active cluster-sender channels to another set of cluster-transmission queues” on page 1026.

  Choose this option if you want the queue manager to make the switch for you.

  An alternative way to describe this option is to say the queue manager switches a cluster-sender channel without you forcing the channel to stop. You do have the option of forcing the channel to stop, and then starting the channel, to make the switch happen sooner. The switch starts when the channel starts, and runs while the channel is running, which is different to option 2. In option 2, the switch takes place when the channel is stopped.

  If you choose this option by letting the switch happen automatically, the switching process starts when a cluster-sender channel starts. If the channel is not stopped, it starts after it becomes inactive, if there is a message to process. If the channel is stopped, start it with the `START CHANNEL` command.

  The switch process completes as soon as there are no messages left for the cluster-sender channel on the transmission queue the channel was serving. As soon as that is the case, newly arrived messages for the cluster-sender channel are stored directly on the new transmission queue. Until then, messages are stored on the old transmission queue, and the switching process transfers messages from the old transmission queue to the new transmission queue. The cluster-sender channel forwards messages from the new cluster transmission queue during the whole switching process.

  When the switch process completes depends on the state of the system. If you are making the changes in a maintenance window, assess beforehand whether the switching process will complete in time. Whether it will complete in time depends on whether the number of messages that are awaiting transfer from the old transmission queue reaches zero.

The advantage of the first method is it is automatic. A disadvantage is that if the time to make the configuration changes is limited to a maintenance window, you must be confident that you can control the system to complete the switch process inside the maintenance window. If you cannot be sure, option 2 might be a better choice.

- Option 2: Make the changes manually; see “Switching a stopped cluster-sender channel to another cluster transmission queue” on page 1027.

  Choose this option if you want to control the entire switching process manually, or if you want to switch a stopped or inactive channel. It is a good choice, if you are switching a few cluster-sender channels, and you want to do the switch during a maintenance window.

  An alternative description of this option is to say that you switch the cluster-sender channel, while the cluster-sender channel is stopped.

  If you choose this option you have complete control over when the switch takes place.

  You can be certain about completing the switching process in a fixed amount of time, within a maintenance window. The time the switch takes depends on how many messages have to be transferred from one transmission queue to the other. If messages keep arriving, it might take a time for the process to transfer all the messages.

  You have the option of switching the channel without transferring messages from the old transmission queue. The switch is “instant”.

  When you restart the cluster-sender channel, it starts processing messages on the transmission queue you newly assigned to it.

  The advantage of the second method is you have control over the switching process. The disadvantage is that you must identify the cluster-sender channels to be switched, run the necessary commands, and resolve any in-doubt channels that might be preventing the cluster-sender channel stopping.

Related tasks:

“Switching active cluster-sender channels to another set of cluster-transmission queues” on page 1026
This task gives you three options for switching active cluster-sender channels. One option is to let the queue manager make the switch automatically, which does not affect running applications. The other options are to stop and start channels manually, or to restart the queue manager.

“Switching a stopped cluster-sender channel to another cluster transmission queue” on page 1027

Related information:

“How the process to switch cluster-sender channel to a different transmission queue works” on page 900

Switching active cluster-sender channels to another set of cluster-transmission queues:

This task gives you three options for switching active cluster-sender channels. One option is to let the queue manager make the switch automatically, which does not affect running applications. The other options are to stop and start channels manually, or to restart the queue manager.

Before you begin

Change the cluster transmission queue configuration. You can change the **DEFCLXQ** queue manager attribute, or add or modify the **CLCHNAME** attribute of transmission queues.

If you reduce the number of messages the switching process has to transfer to the new transmission queue, switching completes more quickly. Read "How the process to switch cluster-sender channel to a different transmission queue works" on page 900 for the reasons for trying to empty the transmission queue before proceeding any further.

About this task

Use the steps in the task as a basis for working out your own plan for making cluster-transmission queue configuration changes.

Procedure

1. Optional: Record the current channel status

   Make a record of the status of current and saved channels that are serving cluster transmission queues. The following commands display the status associated with system cluster transmission queues. Add your own commands to display the status associated with cluster-transmission queues that you have defined. Use a convention, such as `XMITQ.`*`ChannelName`*, to name cluster transmission queues that you define to make it easy to display the channel status for those transmission queues.

   ```
   DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY CHSTATUS(*) SAVED WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   ```
2. Switch transmission queues.
   - Do nothing. The queue manager switches cluster-sender channels when they restart after being stopped or inactive.

     Choose this option if you have no rules or concerns about altering a queue manager configuration. Running applications are not affected by the changes.
   - Restart the queue manager. All cluster-sender channels are stopped and restarted automatically on demand.

     Choose this option to initiate all the changes immediately. Running applications are interrupted by the queue manager as it shuts down and restarts.
   - Stop individual cluster-sender channels and restart them.

     Choose this option to change a few channels immediately. Running applications experience a short delay in message transfer between your stopping and starting the message channel again. The cluster-sender channel remains running, except during the time you stopped it. During the switch process messages are delivered to the old transmission queue, transferred to the new transmission queue by the switching process, and forwarded from the new transmission queue by the cluster-sender channel.
3. Optional: Monitor the channels as they switch

   Display the channel status and the transmission queue depth during the switch. The following example display the status for system cluster transmission queues.

   ```
   DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY CHSTATUS(*) SAVED WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY QUEUE('SYSTEM.CLUSTER.TRANSMIT.*') CURDEPTH
   ```

4. Optional: Monitor the messages "AMQ7341 `The transmission queue for channel` *ChannelName* `switched from queue` *QueueName* `to` *QueueName*" that are written to the queue manager error log.

Switching a stopped cluster-sender channel to another cluster transmission queue:

Before you begin

You might make some configuration changes, and now want to make them effective without starting the cluster-sender channels that are affected. Alternatively, you make the configuration changes you require as one of the steps in the task.

If you reduce the number of messages the switching process has to transfer to the new transmission queue, switching completes more quickly. Read "How the process to switch cluster-sender channel to a different transmission queue works" on page 900 for the reasons for trying to empty the transmission queue before proceeding any further.

About this task

This task switches the transmission queues served by stopped or inactive cluster-sender channels. You might do this task because a cluster-sender channel is stopped, and you want to switch its transmission queue immediately. For example, for some reason a cluster-sender channel is not starting, or has some other configuration problem. To resolve the problem, you decide to create a cluster-sender channel, and associate the transmission queue for the old cluster-sender channel with the new cluster-sender channel you defined.

A more likely scenario is you want to control when reconfiguration of cluster transmission queues is performed. To fully control the reconfiguration, you stop the channels, change the configuration, and then switch the transmission queues.

Procedure

1. Stop the channels that you intend to switch
   a. Stop any running or inactive channels that you intend to switch. Stopping an inactive cluster-sender channel prevents it starting while you are making configuration changes.
      ```
      STOP CHANNEL(ChannelName) MODE(QUIESCSE) STATUS(STOPPED)
      ```
2. Optional: Make the configuration changes.

   For example, see "Clustering: Example configuration of multiple cluster transmission queues" on page 1017.
3. Switch the cluster-sender channels to the new cluster transmission queues.
   ```
   runswchl -m QmgrName -c ChannelName
   ```
   The **runswchl** command transfers any messages on the old transmission queue to the new transmission queue. When the number of messages on the old transmission queue for this channel reaches zero, the switch is completed. The command is synchronous. The command writes progress messages to the window during the switching process.

   During the transfer phase existing and new messages destined for the cluster-sender channel are transferred in order to the new transmission queue.

   Because the cluster-sender channel is stopped, the messages build up on the new transmission queue. Contrast the stopped cluster-sender channel, to step 2 on page 1026 in "Switching active cluster-sender channels to another set of cluster-transmission queues" on page 1026. In that step, the cluster-sender channel is running, so messages do not necessarily build up on the new transmission queue.
4. Optional: Monitor the channels as they switch

   In a different command window, display the transmission queue depth during the switch. The following example display the status for system cluster transmission queues.

```
DISPLAY QUEUE('SYSTEM.CLUSTER.TRANSMIT.*') CURDEPTH
```

5. Optional: Monitor the messages "AMQ7341 `The transmission queue for channel` *`ChannelName`* `switched from queue` *`QueueName`* `to` *`QueueName`*" that are written to the queue manager error log.
6. Restart the cluster-sender channels that you stopped.

   The channels do not start automatically, as you stopped them, placing them into `STOPPED` status.

   ```
   START CHANNEL(ChannelName)
   ```

Related information:

runswchl

RESOLVE CHANNEL

STOP CHANNEL

Clustering: Migration and modification best practices

This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

1. "Moving objects in a cluster" (Best practices for moving objects around inside a cluster, without installing any fix packs or new versions of IBM WebSphere MQ).
2. "Upgrades and maintenance installations" on page 1029 (Best practices for keeping a working cluster architecture up and running, while applying maintenance or upgrades and testing the new architecture).

Moving objects in a cluster

Applications and their queues

When you must move a queue instance hosted on one queue manager to be hosted on another, you can work with the workload balancing parameters to ensure a smooth transition.

Create an instance of the queue where it is to be newly hosted, but use cluster workload balancing settings to continue sending messages to the original instance until your application is ready to switch. This is achieved with the following steps:

1. Set the **`CLWLRANK`** property of the existing queue to a high value, for example five.
2. Create the new instance of the queue and set its **`CLWLRANK`** property to zero.
3. Complete any further configuration of the new system, for example deploy and start consuming applications against the new instance of the queue.
4. Set the **`CLWLRANK`** property of the new queue instance to be higher than the original instance, for example nine.
5. Allow the original queue instance to process any queued messages in the system and then delete the queue.

Moving entire queue managers

If the queue manager is staying on the same host, but the IP address is changing, then the process is as follows:

- DNS, when used correctly, can help simplify the process. For information about using DNS by setting the Connection name (CONNAME) channel attribute, see ALTER CHANNEL .
- If moving a full repository, ensure that you have at least one other full repository which is running smoothly (no problems with channel status for example) before making changes.
- Suspend the queue manager using the SUSPEND QMGR command to avoid traffic buildup.
- Modify the IP address of the computer. If your CLUSRCVR channel definition uses an IP address in the CONNAME field, modify this IP address entry. The DNS cache might need to be flushed through to ensure that updates are available everywhere.
- When the queue manager reconnects to the full repositories, channel auto-definitions automatically resolve themselves.

- If the queue manager hosted a full repository and the IP address changes, it is important to ensure that partials are switched over as soon as possible to point any manually defined CLUSSDR channels to the new location. Until this switch is carried out, these queue managers might be able to only contact the remaining (unchanged) full repository, and warning messages might be seen regarding the incorrect channel definition.
- Resume the queue manager using the ../com.ibm.mq.ref.adm.doc/q086590_.dita command.

If the queue manager must be moved to a new host, it is possible to copy the queue manager data and restore from a backup. This process is not recommended however, unless there are no other options; it might be better to create a queue manager on a new machine and replicate queues and applications as described in the previous section. This situation gives a smooth rollover/rollback mechanism.

If you are determined to move a complete queue manager using backup, follow these best practices:

- Treat the whole process as a queue manager restore from backup, applying any processes you would usually use for system recovery as appropriate for your operating system environment.
- Use the **REFRESH CLUSTER** command after migration to discard all locally held cluster information (including any auto-defined channels that are in doubt), and force it to be rebuilt.

  Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

When creating a queue manager and replicating the setup from an existing queue manager in the cluster (as described previously in this topic), never treat the two different queue managers as actually being the same. In particular, do not give a new queue manager the same queue manager name and IP address. Attempting to 'drop in' a replacement queue manager is a frequent cause of problems in IBM WebSphere MQ clusters. The cache expects to receive updates including the **QMID** attribute, and state can be corrupted.

If two different queue managers are accidentally created with the same name, it is recommended to use the RESET CLUSTER **QMID** command to eject the incorrect entry from the cluster.

Upgrades and maintenance installations

Avoid the "big bang scenario" (for example, stopping all cluster and queue manager activity, applying all upgrades and maintenance to all queue managers, then starting everything at the same time): Clusters are designed to still work with multiple versions of queue manager coexisting, so a well-planned, phased maintenance approach is recommended.

Have a backup plan:

- On z/OS, have you applied backwards migration PTFs?
- Have you taken backups?
- Avoid using new cluster functionality immediately: Wait until you are sure that all the queue managers are upgraded to the new level, and are certain that you are not going to roll any of them back. Using new cluster function in a cluster where some queue managers are still at an earlier level can lead to undefined behavior. For example, in the move to IBM WebSphere MQ Version 7.1 from IBM WebSphere MQ Version 6.0, if a queue manager defines a cluster topic, IBM WebSphere MQ Version 6.0 queue managers will not understand the definition or be able to publish on this topic.

Migrate full repositories first. Although they can forward information that they do not understand, they cannot persist it, so it is not the recommended approach unless absolutely necessary. For more information, see Queue manager cluster migration.

Related concepts:

"Clustering: Using REFRESH CLUSTER best practices"
You use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster. You should not need to use this command, except in exceptional circumstances. If you do need to use it, there are special considerations about how you use it. This information is a guide based on testing and feedback from customers.

Clustering: Using REFRESH CLUSTER best practices

You use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster. You should not need to use this command, except in exceptional circumstances. If you do need to use it, there are special considerations about how you use it. This information is a guide based on testing and feedback from customers.

Only run REFRESH CLUSTER if you really need to do so

The IBM WebSphere MQ cluster technology ensures that any change to the cluster configuration, such as a change to a clustered queue, automatically becomes known to any member of the cluster that needs to know the information. There is no need for further administrative steps to be taken to achieve this propagation of information.

If such information does not reach the queue managers in the cluster where it is required, for example a clustered queue is not known by another queue manager in the cluster when an application attempts to open it for the first time, it implies a problem in the cluster infrastructure. For example, it is possible that a channel cannot be started between a queue manager and a full repository queue manager. Therefore, any situation where inconsistencies are observed must be investigated. If possible, resolve the situation without using the **REFRESH CLUSTER** command.

In rare circumstances that are documented elsewhere in this product documentation, or when requested by IBM support, you can use the **REFRESH CLUSTER** command to discard all locally held information about a cluster and rebuild that information from the full repositories in the cluster.

Refreshing in a large cluster can affect performance and availability of the cluster

Use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, for example by creating a sudden increase in work for the full repositories as they process the repropagation of queue manager cluster resources. If you are refreshing in a large cluster (that is, many hundreds of queue managers) you should avoid use of the command in day-to-day work if possible and use alternative methods to correct specific inconsistencies. For example, if a cluster queue is not being correctly propagated across the cluster, an initial investigation technique of updating the clustered queue definition, such as altering its description, repropagates the queue configuration across the cluster. This process can help to identify the problem and potentially resolve a temporary inconsistency.

If alternative methods cannot be used, and you have to run **REFRESH CLUSTER** in a large cluster, you should do so at off-peak times or during a maintenance window to avoid impact on user workloads. You should also avoid refreshing a large cluster in a single batch, and instead stagger the activity as explained in "Avoid performance and availability issues when cluster objects send automatic updates."

Avoid performance and availability issues when cluster objects send automatic updates

After a new cluster object is defined on a queue manager, an update for this object is generated every 27 days from the time of definition, and sent to every full repository in the cluster and onwards to any other interested queue managers. When you issue the **REFRESH CLUSTER** command to a queue manager, you reset the clock for this automatic update on all objects defined locally in the specified cluster.

If you refresh a large cluster (that is, many hundreds of queue managers) in a single batch, or in other circumstances such as recreating a system from configuration backup, after 27 days all of those queue managers will re-advertise all of their object definitions to the full repositories at the same time. This could again cause the system to run significantly slower, or even become unavailable, until all the updates have completed. Therefore, when you have to refresh or recreate multiple queue managers in a large cluster, you should stagger the activity over several hours, or several days, so that subsequent automatic updates do not regularly impact system performance.

The system cluster history queue

When a **REFRESH CLUSTER** is performed, the queue manager takes a snapshot of the cluster state before the refresh and stores it on the SYSTEM.CLUSTER.HISTORY.QUEUE (SCHQ) if it is defined on the queue manager. This snapshot is for IBM service purposes only, in case of later problems with the system. The SCHQ is defined by default on distributed queue managers on startup. For z/OS migration, the SCHQ must be manually defined. Messages on the SCHQ expire after three months.

Related information:

REFRESH CLUSTER considerations for publish/subscribe clusters
Issuing the **REFRESH CLUSTER** command results in the queue manager temporarily discarding locally held information about a cluster, including any cluster topics and their associated proxy subscriptions.

Clustering: Availability, multi instance, and disaster recovery

This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

IBM WebSphere MQ Clustering itself is not a High Availability solution, but in some circumstances it can be used to improve the availability of services using IBM WebSphere MQ, for example by having multiple instances of a queue on different queue managers. This section gives guidance on ensuring that the IBM WebSphere MQ infrastructure is as highly available as possible so that it can be used in such an architecture.

Availability of cluster resources

The reason for the usual recommendation to maintain two full repositories is that the loss of one is not critical to the smooth running of the cluster. Even if both become unavailable, there is a 60 day grace period for existing knowledge held by partial repositories, although new or not previously accessed resources (queues for example) are not available in this event.

Using clusters to improve application availability

A cluster can help in designing highly available applications (for example a request/response type server application), by using multiple instances of the queue and application. If needed, priority attributes can give preference to the 'live' application unless a queue manager or channel for example become unavailable. This is powerful for switching over quickly to continue processing new messages when a problem occurs.

However, messages which were delivered to a particular queue manager in a cluster are held only on that queue instance, and are not available for processing until that queue manager is recovered. For this reason, for true data high availability you might want to consider other technologies such as multi-instance queue managers.

Multi-instance queue managers

Software High Availability (multi instance) is the best built-in offering for keeping your existing messages available. See “Using WebSphere MQ with high availability configurations” on page 1040, “Create a multi-instance queue manager” on page 1071, and the following section for more information. Any queue manager in a cluster may be made highly available using this technique, as long as all queue managers in the cluster are running at least IBM WebSphere MQ Version 7.0.1. If any queue managers in the cluster are at previous levels, they might lose connectivity with the multi-instance queue managers if they fail over to a secondary IP.

As discussed previously in this topic, as long as two full repositories are configured, they are almost by their nature highly available. If you need to, IBM WebSphere MQ software High Availability / multi-instance queue managers can be used for full repositories. There is no strong reason to use these methods, and in fact for temporary outages these methods might cause additional performance cost during the failover. Using software HA instead of running two full repositories is discouraged because in the event of a single channel outage, for example, it would not necessarily fail over, but might leave partial repositories unable to query for cluster resources.

Disaster recovery

Disaster recovery, for example recovering from when the disks storing a queue manager's data becomes corrupt, is difficult to do well; IBM WebSphere MQ can help, but it cannot do it automatically. The only 'true' disaster recovery option in IBM WebSphere MQ (excluding any operating system or other underlying replication technologies) is restoration from a backup. There are some cluster specific points to consider in these situations:

- Take care when testing disaster recovery scenarios. For example, if testing the operation of backup queue managers, be careful when bringing them online in the same network as it is possible to accidentally join the live cluster and start 'stealing' messages by hosting the same named queues as in the live cluster queue managers.
- Disaster recovery testing must not interfere with a running live cluster. Techniques to avoid interference include:
  - Complete network separation or separation at the firewall level.
  - Not issuing live SSL certificate to the disaster recovery system until, or unless, an actual disaster recovery scenario occurs.
- When restoring a backup of a queue manager in the cluster it is possible that the backup is out of sync with the rest of the cluster. The **REFRESH CLUSTER** command can resolve updates and synchronize with the cluster but the **REFRESH CLUSTER** command must be used as a last resort. See "Clustering: Using REFRESH CLUSTER best practices" on page 1030. Review any in-house process documentation and IBM WebSphere MQ documentation to see whether a simple step was missed before resorting to using the command.
- As for any recovery, applications must deal with replay and loss of data. It must be decided whether to clear the queues down to a known state, or if there is enough information elsewhere to manage replays.

Clustering: Monitoring

This topic provides guidance for planning and administering IBM WebSphere MQ clusters. This information is a guide based on testing and feedback from customers.

Monitoring application messages in the cluster

Typically, all cluster messages that leave the queue manager pass through the SYSTEM.CLUSTER.TRANSMIT.QUEUE, irrespective of which cluster sender channel is being used to transmit the message. Each channel is draining messages targeted for that channel in parallel with all other cluster sender channels. A growing build-up of messages on this queue can indicate a problem with one or more channels and must be investigated:

- The depth of the queue must be monitored appropriately for the cluster design.
- The following command returns all channels that have more than one message that is waiting on the transmit queue:

  ```
  DIS CHSTATUS(*) WHERE(XQMSGSA GT 1)
  ```

  With all cluster messages on a single queue, it is not always easy to see which channel has problems when it begins to fill up. Using this command is an easy way to see which channel is responsible.

You can configure a cluster queue manager to have multiple transmission queues. If you change the queue manager attribute DEFCLXQ to CHANNEL, every cluster-sender channel is associated with a different cluster transmit queue. Alternatively you can configure separate transmission queues manually. To display all the cluster transmit queues that are associated with cluster-sender channels, run the command:

```
DISPLAY CLUSQMGR (qmgrName) XMITQ
```

Define cluster transmission queues so that they follow the pattern of having the fixed stem of the queue name on the left. You can then query the depth of all the cluster transmission queues returned by the **DISPLAY CLUSMGR** command, by using a generic queue name:

```
DISPLAY QUEUE (qname*) CURDEPTH
```

Monitoring control messages in the cluster

The SYSTEM.CLUSTER.COMMAND.QUEUE queue is used for processing all cluster control messages for a queue manager, either generated by the local queue manager or sent to this queue manager from other queue managers in the cluster. When a queue manager is correctly maintaining its cluster state, this queue tends toward zero. There are situations where the depth of messages on this queue can temporarily grow however:

- Having lots of messages on the queue indicates churn in the cluster state.
- When making significant changes, allow the queue to settle in between those changes. For example, when moving repositories, allow the queue to reach zero before moving the second repository.

While a backlog of messages exists on this queue, updates to the cluster state or cluster-related commands are not processed. If messages are not being removed from this queue for a long time, further investigation is required, initially through inspection of the queue manager error logs which might explain the process that is causing this situation.

The SYSTEM.CLUSTER.REPOSITORY.QUEUE holds the cluster repository cache information as a number of messages. It is usual for messages to always exist on this queue, and more for larger clusters. Therefore, the depth of messages on this queue is not an issue for concern.

Monitoring logs

Problems that occur in the cluster might not show external symptoms to applications for many days (and even months) after the problem originally occurs due to the caching of information and the distributed nature of clustering. However, the original problem is often reported in the IBM WebSphere MQ error logs. For this reason, it is vital to actively monitor these logs for any messages written that relate to clustering. These messages must be read and understood, with any action taken where necessary.

For example: A break in communications with a queue manager in a cluster can result in knowledge of certain cluster resources that are being deleted due to the way that clusters regularly revalidate the cluster resources by republishing the information. A warning of such an event potentially occurring is reported by the message AMQ9465. This message indicates that the problem needs to be investigated.

Special considerations for load balancing

When the cluster load balances between two or more instances of a queue, consuming applications must be processing messages on each of the instances. If one or more of those consuming applications terminates or stops processing messages, it is possible that clustering might continue to send messages to those instances of the queue. In this situation, those messages are not processed until the applications are functioning correctly again. For this reason the monitoring of the applications is an important part of the solution and action must be taken to reroute messages in that situation. An example of a mechanism to automate such monitoring can be found in this sample: The Cluster Queue Monitoring sample program (AMQSCLM).

Availability, recovery and restart

Make your applications highly available by maintaining queue availability if a queue manager fails, and recover messages after server or storage failure.

Improve client application availability by using client reconnection to switch a client automatically between a group of queue managers, or to the new active instance of a multi-instance queue manager after a queue manager failure. Automatic client reconnect is not supported by WebSphere MQ classes for Java.

On Windows, UNIX, Linux and IBM i platforms deploy server applications to a multi-instance queue manager, which is configured to run as a single queue manager on multiple servers; if the server running the active instance fails, execution is automatically switched to a standby instance of the same queue manager on a different server. If you configure server applications to run as queue manager services, they are restarted when a standby instance becomes the actively running queue manager instance.

You can configure WebSphere MQ as part of a platform-specific clustering solution such as Microsoft Cluster Server, or PowerHA for AIX (formerly HACMP on AIX) and other UNIX and Linux clustering solutions.

Another way to increase server application availability is to deploy server applications to multiple computers in a queue manager cluster.

A messaging system ensures that messages entered into the system are delivered to their destination. WebSphere MQ can trace the route of a message as it moves from one queue manager to another using the **dspmqrte** command. If a system fails, messages can be recovered in various ways depending on the type of failure, and the way a system is configured.

WebSphere MQ ensures that messages are not lost by maintaining recovery logs of the activities of the queue managers that handle the receipt, transmission, and delivery of messages. It uses these logs for three types of recovery:

1. *Restart recovery*, when you stop WebSphere MQ in a planned way.
2. *Failure recovery*, when a failure stops WebSphere MQ.
3. *Media recovery*, to restore damaged objects.

In all cases, the recovery restores the queue manager to the state it was in when the queue manager stopped, except that any in-flight transactions are rolled back, removing from the queues any updates that were in-flight at the time the queue manager stopped. Recovery restores all persistent messages; nonpersistent messages might be lost during the process.

Automatic client reconnection

You can make your client applications reconnect automatically, without writing any additional code, by configuring a number of components.

Automatic client reconnection is *inline*. The connection is automatically restored at any point in the client application program, and the handles to open objects are all restored.

In contrast, manual reconnection requires the client application to re-create a connection using MQCONN or MQCONNX, and to reopen objects. Automatic client reconnection is suitable for many, but not all client applications.

Table 129 on page 1035 lists the earliest release of WebSphere MQ client support that must be installed on a client workstation. You must upgrade client workstations to one of these levels for an application to use automatic client reconnection. Table 130 on page 1035 lists other requirements to enable automatic client reconnection.

With program access to reconnection options, a client application can set reconnection options. Except for JMS and XMS clients, if a client application has access to reconnection options, it can also create an event handler to handle reconnection events.

An existing client application might be able to benefit from reconnection support, without recompilation and linking:

- For a non-JMS client, set the `mqclient.ini` environment variable `DefRecon` to set reconnection options. Use a CCDT to connect to a queue manager. If the client is to connect to a multi-instance queue manager, provide the network addresses of the active and standby queue manager instances in the CCDT.
- For a JMS client, set the reconnection options in the connection factory configuration. When you use the WebSphere MQ Resource adapter or a JMS Client that is integrated in a Java EE Environment, automatic client reconnection might not be available. There are restrictions in some of the managed environments, for more information see WebSphere MQ automatic client reconnection with WebSphere MQ classes for JMS.

Table 129. Supported clients

Client interface	Client	Program access to reconnection options	Reconnection support
Messaging APIs	C, C++, COBOL, Unmanaged Visual Basic, XMS (Unmanaged XMS on Windows)	7.0.1	7.0.1
	JMS (JSE, and Java EE client container and managed containers)	7.0.1.3	7.0.1.3
	WebSphere MQ classes for Java	Not supported	Not supported
	Managed XMS and managed .NET clients: C#, Visual Basic	7.1	7.1
Other APIs	Windows Communication Foundation (Unmanaged [1])	Not supported	7.0.1
	Windows Communication Foundation (Managed [1])	Not supported	Not supported
	Axis 1	Not supported	Not supported
	Axis 2	Not supported	7.0.1.3
	HTTP (web 2.0)	Not supported	7.0.1.3

1. Set managed or unmanaged mode in the WCF binding configuration.

Automatic reconnection has the following configuration requirements:

Table 130. Automatic reconnection configuration requirements

Component	Requirement	Effect of not meeting requirement
WebSphere MQ MQI client installation	See Table 129	`MQRC_OPTIONS_ERROR`
WebSphere MQ Server installation	Level 7.0.1	`MQRC_OPTIONS_ERROR`
Channel	`SHARECNV > 0`	`MQRC_ENVIRONMENT_ERROR`
Application environment	Must be threaded	`MQRC_ENVIRONMENT_ERROR`

Table 130. Automatic reconnection configuration requirements (continued)

Component	Requirement	Effect of not meeting requirement
MQI	One of: • MQCONNX with `MQCNO Options` set to `MQCNO_RECONNECT` or `MQCNO_RECONNECT_Q_MGR`. • `Defrecon=YES|QMGR` in `mqclient.ini` • In JMS set the CLIENTRECONNECTOPTIONS property of the connection factory.	`MQCC_FAILED` when a connection is broken or queue manager ends or fails.

Figure 168 shows the main interactions between components that are involved in client reconnection.

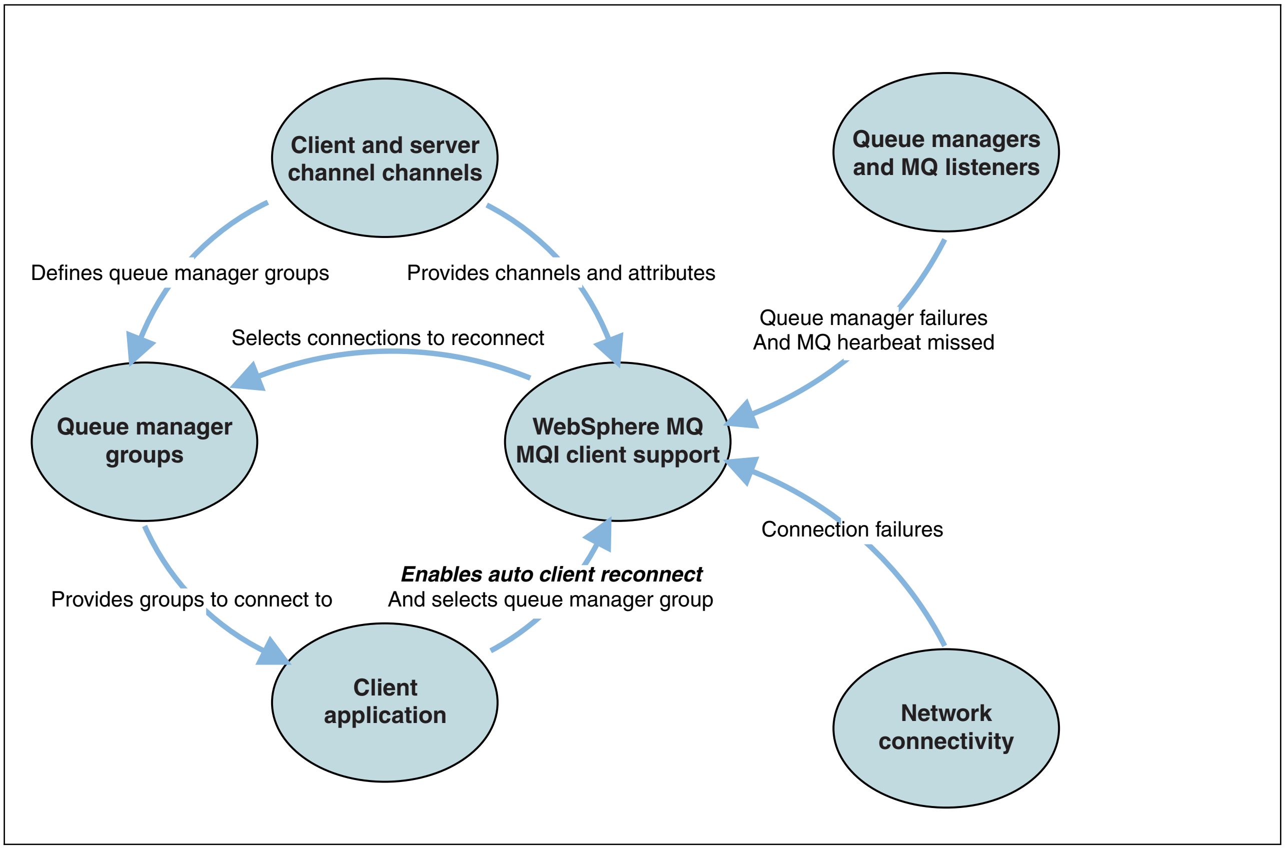

Client and server channel channels
Queue managers and MQ listeners
Defines queue manager groups
Provides channels and attributes
Selects connections to reconnect
Queue manager failures
And MQ hearbeat missed
Queue manager groups
WebSphere MQ MQI client support
Connection failures
Provides groups to connect to
Enables auto client reconnect
And selects queue manager group
Client application
Network connectivity

Figure 168. Automatic client reconnection

Client application

The client application is a WebSphere MQ MQI client.

- By default clients are not automatically reconnected. Enable the automatic client reconnection by setting the MQCONNX `MQCNO Option MQCNO_RECONNECT` or `MQCNO_RECONNECT_Q_MGR`.
- Many applications are written in such a way that they are able to take advantage of auto-reconnection with no additional coding. Enable automatic reconnection for existing programs, without making any coding changes, by setting the `DefRecon` attribute in the channels stanza of the `mqclient.ini` configuration file.
- Use one of these three options:

1. Modify the program so that the logic is unaffected by reconnection. For example, you might have to issue MQI calls within the sync point, and resubmit backed-out transactions.
2. Add an event handler to detect reconnection, and restore the state of the client application when the connection is reestablished.
3. Do not enable auto-reconnection: instead, disconnect the client and issue a new MQCONN or MQCONNX MQI call to find another queue manager instance that is running in the same queue manager group.

For further details about these three options, see "Application recovery" on page 1116.

- Reconnecting to a queue manager of the same name does not guarantee that you have reconnected to the same instance of a queue manager.

  Use an `MQCNO` option `MQCNO_RECONNECT_Q_MGR` , to reconnect to an instance of the same queue manager.
- A client can register an event handler so that it can be informed the state of reconnection. The `MQHCONN` passed in the event handler cannot be used. The following reason codes are provided:

  MQRC_RECONNECTING
  The connection failed, and the system is attempting to reconnect. You receive multiple `MQRC_RECONNECTING` events if multiple reconnect attempts are made.

  MQRC_RECONNECTED
  The reconnection made and all handles successfully reestablished.

  MQRC_RECONNECT_FAILED
  The reconnection was not successful.

  MQRC_RECONNECT_QMID_MISMATCH
  A reconnectable connection specified `MQCNO_RECONNECT_Q_MGR` and the connection attempted to reconnect to a different queue manager.

  MQRC_RECONNECT_Q_MGR_REQD
  An option, such `MQMO_MATCH_MSG_TOKEN` in an MQGET call, was specified in the client program that requires reconnection to the same queue manager.
- A reconnectable client is able to reconnect automatically only *after* connecting. That is, the MQCONNX call itself is not tried again if it fails. For example, if you receive the return code `2543 - MQRC_STANDBY_Q_MGR` from MQCONNX, reissue the call after a short delay.

  MQRC_RECONNECT_INCOMPATIBLE
  This reason code is returned when the application tries to use `MQPMO_LOGICAL_ORDER` (with MQPUT and MQPUT1) or `MQGMO_LOGICAL_ORDER` (with MQGET) when reconnect options are set. The reason for returning the reason code is to make sure that applications never use reconnect in such cases.

  MQRC_CALL_INTERRUPTED
  This reason code is returned when the connection breaks during the execution of Commit call and the client reconnects. An MQPUT of a persistent message outside the sync point also results in the same reason code being returned to the application.

Multi-instance queue managers

Simplify restarting WebSphere MQ MQI client applications, after a multi-instance queue manager has activated its standby instance, by using automatic client reconnection.

The standby instance of a multi-instance queue manager is typically at a different network address to the active instance. Include the network addresses of both the instances in the client connection definition table (CCDT). Either provide a list of network addresses for the **CONNAME** parameter, or define multiple rows for the queue manager in the CCDT.

Commonly, WebSphere MQ MQI clients reconnect to any queue manager in a queue manager group. Sometimes you want a WebSphere MQ MQI client to reconnect only to the same queue manager. It might have an affinity to a queue manager. You can prevent a client from reconnecting to a different queue manager. Set the MQCNO option, MQCNO_RECONNECT_Q_MGR. The WebSphere MQ MQI client fails if it reconnects to a different queue manager. If you set the MQCNO option, MQCNO_RECONNECT_Q_MGR, do not include other queue managers in the same queue manager group. The client returns an error if the queue manager it reconnects to is not the same queue manager as the one it connected to.

Queue manager groups

You can select whether the client application always connects and reconnects to a queue manager of the same name, to the same queue manager, or to any of a set of queue managers that are defined with the same QMNAME value in the client connection table.

- The queue manager *name* attribute, QMNAME , in the client channel definition is the name of a queue manager group.
- In your client application, if you set the value of the MQCONN or MQCONNX QmgrName parameter to a queue manager name, the client connects only to queue managers with that name. If you prefix the queue manager name with an asterisk(*), the client connects to any queue manager in the queue manager group with the same QMNAME value. For a full explanation, see Queue manager groups in the CCDT.

Queue sharing groups

Automatic client reconnection to z/OS queue sharing groups, uses the same mechanisms for reconnection as any other environment. The client will reconnect to the same selection of queue managers as is configured for the original connection. For example, when using the client channel definition table the administrator should ensure that all entries in the table, resolve to the same z/OS queue sharing group.

Client and server channel definitions

Client and server channel definitions define the groups of queue managers a client application can reconnect to. The definitions govern the selection and timing of reconnections, and other factors, such as security; see the related topics. The most relevant channel attributes to consider for reconnection are listed in two groups:

Client connection attributes

Connection affinity (AFFINITY)AFFINITY
Connection affinity.

Client channel weight (CLNTWGHT)CLNTWGHT
Client channel weight.

Connection name (CONNAME)CONNAME
Connection information.

Heartbeat interval (HBINT)HBINT
Heartbeat interval. Set the heartbeat interval on the server connection channel.

Keepalive Interval (KAINT)KAINT
Keepalive interval. Set the keepalive interval on the server connection channel.

Note that KAINT applies to z/OS only.

Queue manager name (QMNAME)QMNAME
Queue manager name.

Server connection attributes

Heartbeat interval (HBINT)HBINT

Heartbeat interval. Set the heartbeat interval on the client connection channel.

Keepalive Interval (KAINT)KAINT

Keepalive interval. Set the keepalive interval on the client connection channel.

Note that KAINT applies to z/OS only.

KAINT is a network layer heartbeat, and HBINT is a WebSphere MQ heartbeat between the client and the queue manager. Setting these heartbeats to a shorter time serves two purposes:

1. By simulating activity on the connection, network layer software that is responsible for closing inactive connections is less likely to shut down your connection.
2. If the connection is shut down, the delay before the broken connection is detected, is shortened.

The default TCP/IP keepalive interval is two hours. Consider setting the KAINT and HBINT attributes to a shorter time. Do not assume that the normal behavior of a network suits the needs of automatic reconnection. For example, some firewalls can shut down an inactive TCP/IP connection after as little as 10 minutes.

Network connectivity

Only network failures that are passed to the WebSphere MQ MQI client by the network, are handled by the automatic reconnection capability of the client.

- Reconnections performed automatically by the transport are invisible to WebSphere MQ.
- Setting HBINT helps to deal with network failures that are invisible to WebSphere MQ.

Queue managers and WebSphere MQ listeners

Client reconnection is triggered by server failure, queue manager failure, network connectivity failure, and by an administrator switching over to another queue manager instance.

- If you are using a multi-instance queue manager, an additional cause of client reconnection occurs when you switch control from the active queue manager instance to a standby instance.
- Ending a queue manager using the default **endmqm** command, does not trigger automatic client reconnection. Add the -r option on the **endmqm** command to request automatic client reconnection, or the -s option to transfer to a standby queue manager instance after shutting down.

WebSphere MQ MQI client automatic reconnection support

If you use the automatic client reconnection support in the WebSphere MQ MQI client, the client application automatically reconnects and continues processing without you issuing an MQCONN or MQCONNX MQI call to reconnect to the queue manager.

- Automatic client reconnection is triggered by one of the following occurrences:
  - queue manager failure
  - ending a queue manager and specifying the -r, reconnect, option on the **endmqm** command
- The MQCONNX MQCNO options control whether you have enabled the automatic client reconnection. The options are described in Reconnection options.
- Automatic client reconnection issues MQI calls on behalf of your application to restore the connection handle and the handles to other open objects, so that your program can resume normal processing after it has processed any MQI errors that resulted from the broken connection. See “Recovery of an automatically reconnected client” on page 1118.
- If you have written a channel exit program for the connection, the exit receives these additional MQI calls.
- You can register a reconnection event handler, which is triggered when reconnection begins and when it finishes.

Although reconnection takes no more than a minute, reconnection can take longer because a queue manager might have numerous resources to manage. During this time, a client application might be holding locks that do not belong to WebSphere MQ resources. There is a timeout value you can configure to limit the time a client waits for reconnection. The value (in seconds) is set in the `mqclient.ini` file.

```
Channels:
    MQReconnectTimeout = 1800
```

No reconnection attempts are made after the timeout has expired. When the system detects that the timeout has expired it returns a `MQRC_RECONNECT_FAILED` error.

Console message monitoring

There are a number of information messages issued by the queue manager or channel initiator that should be considered particularly significant. These messages do not in themselves indicate a problem, but may be useful in tracking because they do indicate a potential issue which potentially needs addressing.

The presence of this message can also indicate that a user application is putting a large number of messages to the page set, which may be a symptom of a larger problem:

- A problem with the user application which PUT's messages, such as an uncontrolled loop.
- A user application which GET's the messages from the queue is no longer functioning.

Using WebSphere MQ with high availability configurations

If you want to operate your WebSphere MQ queue managers in a high availability (HA) configuration, you can set up your queue managers to work either with a high availability manager, such as PowerHA® for AIX (formerly HACMP™) or the Microsoft Cluster Service (MSCS), or with WebSphere MQ multi-instance queue managers.

You need to be aware of the following configuration definitions:

Queue manager clusters
Groups of two or more queue managers on one or more computers, providing automatic interconnection, and allowing queues to be shared among them for load balancing and redundancy.

HA clusters
HA clusters are groups of two or more computers and resources such as disks and networks, connected together and configured in such a way that, if one fails, a high availability manager, such as HACMP (UNIX) or MSCS (Windows) performs a *failover*. The failover transfers the state data of applications from the failing computer to another computer in the cluster and re-initiates their operation there. This provides high availability of services running within the HA cluster. The relationship between WebSphere MQ clusters and HA clusters is described in "Relationship of HA clusters to queue manager clusters" on page 1041.

Multi-instance queue managers
Instances of the same queue manager configured on two or more computers. By starting multiple instances, one instance becomes the active instance and the other instances become standbys. If the active instance fails, a standby instance running on a different computer automatically takes over. You can use multi-instance queue managers to configure your own highly available messaging systems based on WebSphere MQ, without requiring a cluster technology such as HACMP or MSCS. HA clusters and multi-instance queue managers are alternative ways of making queue managers highly available. Do not combine them by putting a multi-instance queue manager in an HA cluster.

Differences between multi-instance queue managers and HA clusters

Multi-instance queue managers and HA clusters are alternative ways to achieve high availability for your queue managers. Here are some points that highlight the differences between the two approaches.

Multi-instance queue managers include the following features:
- Basic failover support integrated into WebSphere MQ
- Faster failover than HA cluster
- Simple configuration and operation
- Integration with WebSphere MQ Explorer

Limitations of multi-instance queue managers include:
- Highly available, high performance networked storage required
- More complex network configuration because queue manager changes IP address when it fails over

HA clusters include the following features:
- The ability to coordinate multiple resources, such as an application server or database
- More flexible configuration options including clusters comprising more than two nodes
- Can failover multiple times without operator intervention
- Takeover of queue manager's IP address as part of the failover

Limitations of HA clusters include:
- Additional product purchase and skills are required
- Disks which can be switched between the nodes of the cluster are required
- Configuration of HA clusters is relatively complex
- Failover is rather slow historically, but recent HA cluster products are improving this
- Unnecessary failovers can occur if there are shortcomings in the scripts that are used to monitor resources such as queue managers

Relationship of HA clusters to queue manager clusters

Queue manager clusters reduce administration and provide load balancing of messages across instances of queue manager cluster queues. They also offer higher availability than a single queue manager because, following a failure of a queue manager, messaging applications can still access surviving instances of a queue manager cluster queue. However, queue manager clusters alone do not provide automatic detection of queue manager failure and automatic triggering of queue manager restart or failover. HA clusters provide these features. The two types of cluster can be used together to good effect.

Using WebSphere MQ with a high availability cluster on UNIX and Linux

You can use WebSphere MQ with a high availability (HA) cluster on UNIX and Linux platforms: for example, PowerHA for AIX (formerly HACMP), Veritas Cluster Server, HP Serviceguard, or a Red Hat Enterprise Linux cluster with Red Hat Cluster Suite.

Before WebSphere MQ Version 7.0.1, SupportPac MC91 was provided to assist in configuring HA clusters. WebSphere MQ Version 7.0.1 provided a greater degree of control than previous versions over where queue managers store their data. This makes it easier to configure queue managers in an HA cluster. Most of the scripts provided with SupportPac MC91 are no longer required, and the SupportPac is withdrawn.

This section introduces “HA cluster configurations” on page 1042, the relationship of HA clusters to queue manager clusters, “WebSphere MQ clients” on page 1042, and “WebSphere MQ operating in an

HA cluster" on page 1043, and guides you through the steps and provides example scripts that you can adapt to configure queue managers with an HA cluster.

Refer to the HA cluster documentation particular to your environment for assistance with the configuration steps described in this section.

HA cluster configurations

In this section the term *node* is used to refer to the entity that is running an operating system and the HA software; "computer", "system" or "machine" or "partition" or "blade" might be considered synonyms in this usage. You can use WebSphere MQ to help set up either standby or takeover configurations, including mutual takeover where all cluster nodes are running WebSphere MQ workload.

A *standby* configuration is the most basic HA cluster configuration in which one node performs work while the other node acts only as standby. The standby node does not perform work and is referred to as idle; this configuration is sometimes called *cold standby*. Such a configuration requires a high degree of hardware redundancy. To economize on hardware, it is possible to extend this configuration to have multiple worker nodes with a single standby node. The point of this is that the standby node can take over the work of any other worker node. This configuration is still referred to as a standby configuration and sometimes as an "N+1" configuration.

A *takeover* configuration is a more advanced configuration in which all nodes perform some work and critical work can be taken over in the event of a node failure.

A *one-sided takeover* configuration is one in which a standby node performs some additional, noncritical and unmovable work. This configuration is similar to a standby configuration but with (noncritical) work being performed by the standby node.

A *mutual takeover* configuration is one in which all nodes are performing highly available (movable) work. This type of HA cluster configuration is also sometimes referred to as "Active/Active" to indicate that all nodes are actively processing critical workload.

With the extended standby configuration or either of the takeover configurations it is important to consider the peak load that might be placed on a node that can take over the work of other nodes. Such a node must possess sufficient capacity to maintain an acceptable level of performance.

Relationship of HA clusters to queue manager clusters

Queue manager clusters reduce administration and provide load balancing of messages across instances of queue manager cluster queues. They also offer higher availability than a single queue manager because, following a failure of a queue manager, messaging applications can still access surviving instances of a queue manager cluster queue. However, queue manager clusters alone do not provide automatic detection of queue manager failure and automatic triggering of queue manager restart or failover. HA clusters provide these features. The two types of cluster can be used together to good effect.

WebSphere MQ clients

WebSphere MQ clients that are communicating with a queue manager that might be subject to a restart or takeover must be written to tolerate a broken connection and must repeatedly attempt to reconnect. WebSphere MQ Version 7 introduced features in the processing of the Client Channel Definition Table (CCDT) that assist with connection availability and workload balancing; however these are not directly relevant when working with a failover system.

The Extended Transactional Client (ETC), which allows a WebSphere MQ MQI client to participate in two-phase transactions, must always connect to the same queue manager. The ETC cannot use techniques such as an IP load balancer to select from a list of queue managers. When you use an HA product, a queue manager maintains its identity (name and address) whichever node it is running on, so the ETC can be used with queue managers that are under HA control.

WebSphere MQ operating in an HA cluster

All HA clusters have the concept of a unit of failover. This is a set of definitions that contains all the resources that make up the highly available service. The unit of failover includes the service itself and all other resources upon which it depends.

HA solutions use different terms for a unit of failover:

- On PowerHA for AIX the unit of failover is called a *resource group*.
- On Veritas Cluster Server it is known as a *service group*.
- On Serviceguard it is called a *package*.

This topic uses the term *resource group* to mean a unit of failover.

The smallest unit of failover for WebSphere MQ is a queue manager. Typically, the resource group containing the queue manager also contains shared disks in a volume group or disk group that is reserved exclusively for use by the resource group, and the IP address that is used to connect to the queue manager. It is also possible to include other WebSphere MQ resources, such as a listener or a trigger monitor in the same resource group, either as separate resources, or under the control of the queue manager itself.

A queue manager that is to be used in an HA cluster must have its data and logs on disks that are shared between the nodes in the cluster. The HA cluster ensures that only one node in the cluster at a time can write to the disks. The HA cluster can use a monitor script to monitor the state of the queue manager.

It is possible to use a single shared disk for both the data and logs that are related to the queue manager. However, it is normal practice to use separate shared file systems so that they can be independently sized and tuned.

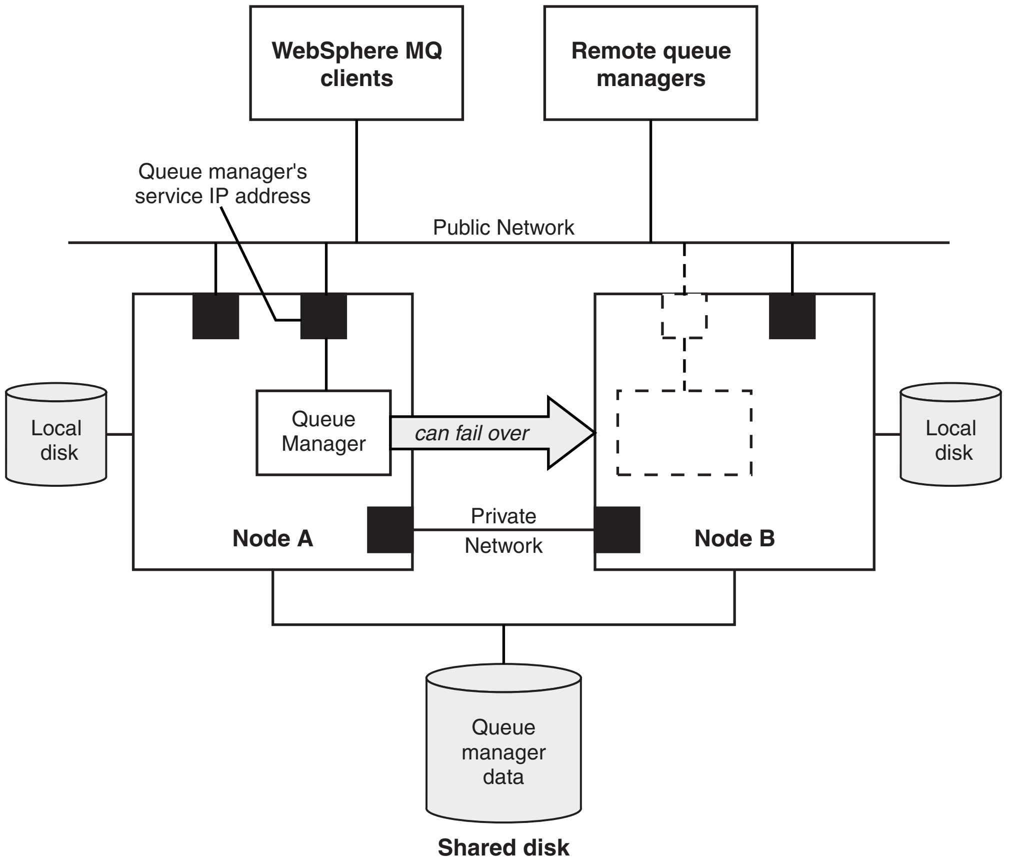


Figure 169. HA cluster

Figure 1 illustrates a HA cluster with two nodes. The HA cluster is managing the availability of a queue manager which has been defined in a resource group. This is an active/passive or cold standby configuration, because only one node, node A, is currently running a queue manager. The queue manager was created with its data and log files on a shared disk. The queue manager has a service IP address which is also managed by the HA cluster. The queue manager depends on the shared disk and its service IP address. When the HA cluster fails the queue manager over from node A to node B, it first moves the queue manager's dependent resources onto node B and then starts the queue manager.

If the HA cluster contains more than one queue manager, your HA cluster configuration might result in two or more queue managers running on the same node after a failover. Each queue manager in the HA cluster must be assigned its own port number, which it uses on whichever cluster node it happens to be active at any particular time.

Generally, the HA cluster runs as the root user. WebSphere MQ runs as the mqm user. Administration of WebSphere MQ is granted to members of the mqm group. Ensure that the mqm user and group both exist on all HA cluster nodes. The user ID and group ID must be consistent across the cluster. Administration of WebSphere MQ by the root user is not allowed; scripts that start, stop, or monitor scripts must switch to the mqm user.

Note: WebSphere MQ must be installed correctly on all nodes; you cannot share the product executable files.

Configuring the shared disks:

A WebSphere MQ queue manager in an HA cluster requires data files and log files to be in common named remote file systems on a shared disk.

To configure the shared disks, complete the following steps:

1. Decide the names of the mount points for the queue manager's file systems. For example, `/MQHA/qmgrname/data` for the queue manager's data files and `/MQHA/qmgrname/log` for its log files.
2. Create a volume group (or disk group) to contain the queue manager's data and log files. This volume group is managed by the high availability (HA) cluster in the same resource group as the queue manager.
3. Create the file systems for the queue manager's data and log files in the volume group.
4. For each node in turn, create the mount points for the file systems and make sure that the file systems can be mounted. The mqm user must own the mount points.

Figure 1 shows a possible layout for a queue manager in an HA cluster. The queue manager's data and log directories are both on the shared disk which is mounted at /MQHA/QM1. This disk is switched between the nodes of the HA cluster when failover occurs so that the data is available wherever the queue manager is restarted. The `mqs.ini` file has a stanza for the QM1 queue manager. The Log stanza in the `qm.ini` file has a value for LogPath.

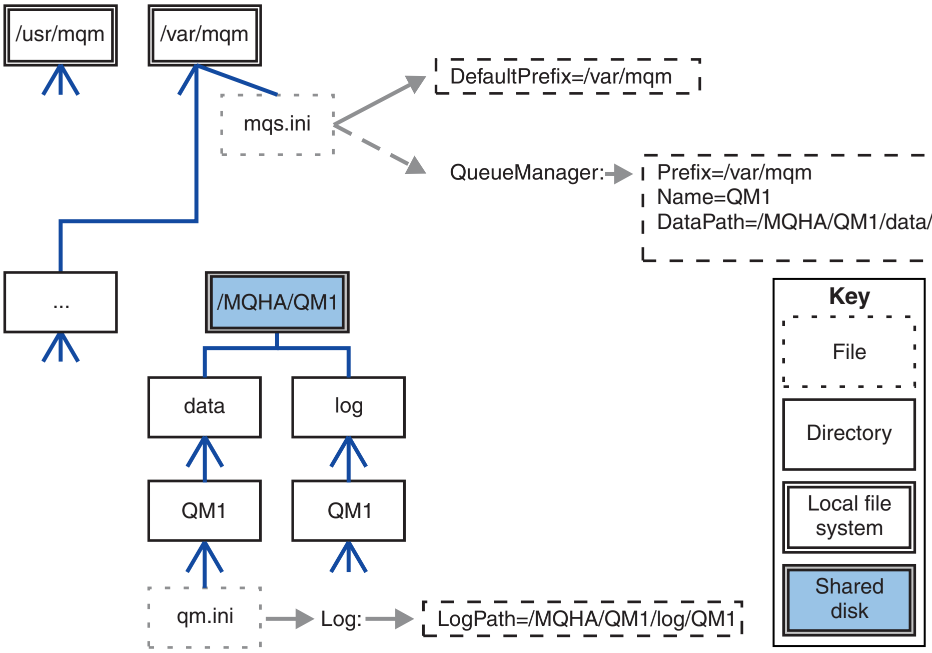


Figure 170. Shared named `data` and `log` directories

Creating a queue manager for use in a high availability (HA) cluster:

The first step towards using a queue manager in a high availability cluster is to create the queue manager on one of the nodes.

To create a queue manager for use in an HA cluster, select one of the nodes in the cluster on which to create the queue manager. On this node complete the following steps:

1. Mount the queue manager's file systems on the node.
2. Create the queue manager by using the **crtmqm** command. For example:
   ```
   crtmqm -md /MQHA/qmgrname/data -ld /MQHA/qmgrname/log qmgrname
   ```
3. Start the queue manager manually by using the **strmqm** command.
4. Complete any initial configuration of the queue manager, such as creating queues and channels, and setting the queue manager to start a listener automatically when the queue manager starts.
5. Stop the queue manager by using the **endmqm** command.
6. Use the **dspmqinf** command to display the **addmqinf** command that you can use in a later task, which is documented in "Adding queue manager configuration information to other nodes in a high availability (HA) cluster":
   ```
   dspmqinf -o command qmgrname
   ```
   where qmgrname is the name of the queue manager.
7. The **addmqinf** command that is displayed will be similar to the following example:
   ```
   addmqinf -sQueueManager -vName=qmgrname -vDirectory=qmgrname \
            -vPrefix=/var/mqm -vDataPath=/MQHA/qmgrname/data/qmgrname
   ```
   Make a careful note of the displayed command.
8. Unmount the queue manager's file systems.

You are now ready to complete the steps described in "Adding queue manager configuration information to other nodes in a high availability (HA) cluster."

Adding queue manager configuration information to other nodes in a high availability (HA) cluster:

You must add the queue manager configuration to the other nodes in the HA cluster.

Before you complete this task, you must have completed the steps in "Creating a queue manager for use in a high availability (HA) cluster."

To add the configuration information for the queue manager to each of other nodes in the HA cluster, complete the following steps on each of the other nodes:

1. Mount the queue manager file systems.
2. Add the queue manager configuration information to the node, either by editing `/var/mqm/mqs.ini` directly, or by issuing the **addmqinf** command that was displayed by the **dspmqinf** command in steps 6 and 7 in "Creating a queue manager for use in a high availability (HA) cluster."
3. Start and stop the queue manager to verify the configuration.

   The commands used to start and stop the queue manager must be issued from the same IBM WebSphere MQ installation as the **addmqinf** command. To start and stop the queue manager from a different installation, you must first set the installation associated with the queue manager using the **setmqm** command. For more information, see setmqm.
4. Unmount the queue manager file systems.

Starting a queue manager under control of a high availability (HA) cluster:

The queue manager is represented in the HA cluster as a resource. The HA cluster must be able to start and stop the queue manager. In most cases you can use a shell script to start the queue manager. You must make these scripts available at the same location on all nodes in the cluster, either using a network filesystem or by copying them to each of the local disks.

Note: Before you restart a failed queue manager, you must disconnect your applications from that instance of the queue manager. If you do not, the queue manager might not restart correctly.

Examples of suitable shell scripts are given here. You can tailor these to your needs and use them to start the queue manager under the control of your HA cluster.

The following shell script is an example of how to switch from the HA cluster user to the mqm user so that the queue manager can be successfully started:

```
#!/bin/ksh

# A simple wrapper script to switch to the mqm user.

su mqm -c name_of_your_script $*
```

The following shell script is an example of how to start a queue manager without making any assumptions about the current state of the queue manager. Note that it uses an extremely abrupt method of ending any processes that belong to the queue manager:

```
#!/bin/ksh
#
# This script robustly starts the queue manager.
#
# The script must be run by the mqm user.

# The only argument is the queue manager name. Save it as QM variable
QM=$1

if [ -z "$QM" ]
then
  echo "ERROR! No queue manager name supplied"
  exit 1
fi

# End any queue manager processes which might be running.

srchstr="( |-m)$QM *.*$"
for process in amqzmuc0 amqzxma0 amqfcxba amqfqpub amqpcsea amqzlaa0 \
               amqzlsa0 runmqchi runmqlsr amqcrsta amqrrmfa amqrmppa \
               amqzfuma amqzdmaa amqzmuf0 amqzmur0 amqzmgr0
 do
  ps -ef | tr "\t" " " | grep $process | grep -v grep | \
    egrep "$srchstr" | awk '{print $2}'| \
       xargs kill -9 > /dev/null 2>&1
done

# It is now safe to start the queue manager.
# The strmqm command does not use the -x flag.
strmqm ${QM}
```

You can modify the script to start other related programs.

Stopping a queue manager under the control of a high availability (HA) cluster:

In most cases, you can use a shell script to stop a queue manager. Examples of suitable shell scripts are given here. You can tailor these to your needs and use them to stop the queue manager under control of your HA cluster.

The following script is an example of how to immediately stop without making assumptions about the current state of the queue manager. The script must be run by the mqm user; it might therefore be necessary to wrap this script in a shell script to switch the user from the HA cluster user to mqm (an example shell script is provided in "Starting a queue manager under control of a high availability (HA) cluster" on page 1047):

```
#!/bin/ksh
#
# The script ends the QM by using two phases, initially trying an immediate
# end with a time-out and escalating to a forced stop of remaining
# processes.
#
# The script must be run by the mqm user.
#
# There are two arguments: the queue manager name and a timeout value.
QM=$1
TIMEOUT=$2

if [ -z "$QM" ]
then
  echo "ERROR! No queue manager name supplied"
  exit 1
fi

if [ -z "$TIMEOUT" ]
then
  echo "ERROR! No timeout specified"
  exit 1
fi

for severity in immediate brutal
do
  # End the queue manager in the background to avoid
  # it blocking indefinitely. Run the TIMEOUT timer
  # at the same time to interrupt the attempt, and try a
  # more forceful version. If the brutal version fails,
  # nothing more can be done here.

  echo "Attempting ${severity} end of queue manager '${QM}'"
  case $severity in

  immediate)
    # Minimum severity of endmqm is immediate which severs connections.
    # HA cluster should not be delayed by clients
    endmqm -i ${QM} &
    ;;

  brutal)
    # This is a forced means of stopping queue manager processes.

    srchstr="( |-m)$QM *.*$"
    for process in amqzmuc0 amqzxma0 amqfcxba amqfqpub amqpcsea amqzlaa0 \
               amqzlsa0 runmqchi runmqlsr amqcrsta amqrrmfa amqrmppa \
               amqzfuma amqzdmaa amqzmuf0 amqzmur0 amqzmgr0
    do
      ps -ef | tr "\t" " " | grep $process | grep -v grep | \
         egrep "$srchstr" | awk '{print $2}'| \
            xargs kill -9 > /dev/null 2>&1
    done
  esac

  TIMED_OUT=yes
  SECONDS=0
  while (( $SECONDS < ${TIMEOUT} ))
  do
   TIMED_OUT=yes
   i=0
   while [ $i -lt 5 ]
   do
     # Check for execution controller termination
     srchstr="( |-m)$QM *.*$"
     cnt=`ps -ef | tr "\t" " " | grep amqzxma0 | grep -v grep | \
       egrep "$srchstr" | awk '{print $2}' | wc -l `
     i=`expr $i + 1`
     sleep 1
     if [ $cnt -eq 0 ]
     then
       TIMED_OUT=no
       break
     fi
   done

   if [ ${TIMED_OUT} = "no" ]
   then
     break
   fi

   echo "Waiting for ${severity} end of queue manager '${QM}'"
   sleep 1
  done # timeout loop

  if [ ${TIMED_OUT} = "yes" ]
  then
    continue        # to next level of urgency
  else
    break           # queue manager is ended, job is done
  fi

done # next phase
```

Monitoring a queue manager:

It is usual to provide a way for the high availability (HA) cluster to monitor the state of the queue manager periodically. In most cases, you can use a shell script for this. Examples of suitable shell scripts are given here. You can tailor these scripts to your needs and use them to make additional monitoring checks specific to your environment.

From WebSphere MQ version 7.1, it is possible to have multiple installations of WebSphere MQ coexisting on a system. For more information about multiple installations, see Multiple installations. If you intend to use the monitoring script across multiple installations, including installations at version 7.1, or higher, you might need to perform some additional steps. If you have a primary installation, or you are using the script with versions earlier than version 7.1, you do not need to specify the*MQ_INSTALLATION_PATH* to use the script. Otherwise, the following steps ensure that the *MQ_INSTALLATION_PATH* is identified correctly:

1. Use the **crtmqenv** command from a version 7.1 installation to identify the correct *MQ_INSTALLATION_PATH* for a queue manager:

   ```
   crtmqenv -m qmname
   ```

   This command returns the correct *MQ_INSTALLATION_PATH* value for the queue manager specified by *qmname*.
2. Run the monitoring script with the appropriate *qmname* and *MQ_INSTALLATION_PATH* parameters.

Note: PowerHA for AIX does not provide a way of supplying a parameter to the monitoring program for the queue manager. You must create a separate monitoring program for each queue manager, that encapsulates the queue manager name. Here is an example of a script used on AIX to encapsulate the queue manager name:

```
#!/bin/ksh
su mqm -c name_of_monitoring_script qmname MQ_INSTALLATION_PATH
```

where *MQ_INSTALLATION_PATH* is an optional parameter that specifies the path to the installation of IBM WebSphere MQ that the queue manager *qmname* is associated with.

The following script is not robust to the possibility that **runmqsc** hangs. Typically, HA clusters treat a hanging monitoring script as a failure and are themselves robust to this possibility.

The script does, however, tolerate the queue manager being in the starting state. This is because it is common for the HA cluster to start monitoring the queue manager as soon as it has started it. Some HA clusters distinguish between a starting phase and a running phase for resources, but it is necessary to configure the duration of the starting phase. Because the time taken to start a queue manager depends on the amount of work that it has to do, it is hard to choose a maximum time that starting a queue manager takes. If you choose a value that is too low, the HA cluster incorrectly assumes that the queue manager failed when it has not completed starting. This could result in an endless sequence of failovers.

This script must be run by the mqm user; it might therefore be necessary to wrap this script in a shell script to switch the user from the HA cluster user to mqm (an example shell script is provided in "Starting a queue manager under control of a high availability (HA) cluster" on page 1047):

```
#!/bin/ksh
#
# This script tests the operation of the queue manager.
#
# An exit code is generated by the runmqsc command:
# 0  => Either the queue manager is starting or the queue manager is running and responds.
#       Either is OK.
# >0 => The queue manager is not responding and not starting.
#
# This script must be run by the mqm user.
QM=$1
MQ_INSTALLATION_PATH=$2

if [ -z "$QM" ]
then
  echo "ERROR! No queue manager name supplied"
  exit 1
fi

if [ -z "$MQ_INSTALLATION_PATH" ]
then
  # No path specified, assume system primary install or MQ level < 7.1.0.0
  echo "INFO: Using shell default value for MQ_INSTALLATION_PATH"
else
  echo "INFO: Prefixing shell PATH variable with $MQ_INSTALLATION_PATH/bin"
  PATH=$MQ_INSTALLATION_PATH/bin:$PATH
fi

# Test the operation of the queue manager. Result is 0 on success, non-zero on error.
echo "ping qmgr" | runmqsc ${QM} > /dev/null 2>&1
pingresult=$?

if [ $pingresult -eq 0 ]
then # ping succeeded

  echo "Queue manager '${QM}' is responsive"
  result=0
```

```
else # ping failed

  # Don't condemn the queue manager immediately, it might be starting.
  srchstr="( |-m)$QM *.*$"
  cnt=`ps -ef | tr "\t" " " | grep strmqm | grep "$srchstr" | grep -v grep \
               | awk '{print $2}' | wc -l`
  if [ $cnt -gt 0 ]
  then
    # It appears that the queue manager is still starting up, tolerate
    echo "Queue manager '${QM}' is starting"
    result=0
  else
    # There is no sign of the queue manager starting
    echo "Queue manager '${QM}' is not responsive"
    result=$pingresult
  fi

fi

exit $result
```

Putting the queue manager under control of the high availability (HA) cluster:

You must configure the queue manager, under control of the HA cluster, with the queue manager's IP address and shared disks.

To define a resource group to contain the queue manager and all of its associated resources, complete the following steps:

1. Create the resource group containing the queue manager, the queue manager's volume or disk group, and the queue manager's IP address. The IP address is a virtual IP address, not the IP address of the computer.
2. Verify that the HA cluster correctly switches the resources between the cluster nodes and is ready to control the queue manager.

Deleting a queue manager from a high availability (HA) cluster node:

You might want to remove a queue manager from a node that is no longer required to run the queue manager.

To remove the queue manager from a node in an HA cluster, complete the following steps:

1. Remove the node from the HA cluster so that the HA cluster will no longer attempt to activate the queue manager on this node.
2. Use the following **rmvmqinf** command to remove the queue manager's configuration information:

   ```
   rmvmqinf qmgrname
   ```

To completely delete the queue manager, use the **dltmqm** command. However, be aware that this completely deletes the queue manager's data and log files. When you have deleted the queue manager, you can use the **rmvmqinf** command to remove remaining configuration information from the other nodes.

Supporting the Microsoft Cluster Service (MSCS)

Introducing and setting up MSCS to support failover of virtual servers.

This information applies to WebSphere MQ for Windows only.

The Microsoft Cluster Service (MSCS) enables you to connect servers into a *cluster*, giving higher availability of data and applications, and making it easier to manage the system. MSCS can automatically detect and recover from server or application failures.

MSCS supports *failover* of *virtual servers*, which correspond to applications, Web sites, print queues, or file shares (including, for example, their disk spindles, files, and IP addresses).

Failover is the process by which MSCS detects a failure in an application on one computer in the cluster, and shuts down the disrupted application in an orderly manner, transfers its state data to the other computer, and reinitiates the application there.

This section introduces MSCS clusters and describes setting up MSCS support in the following sections:

- "Introducing MSCS clusters"
- "Setting up WebSphere MQ for MSCS clustering" on page 1053

Then tells you how to configure WebSphere MQ for MSCS clustering, in the following sections:

- "Creating a queue manager for use with MSCS" on page 1056
- "Moving a queue manager to MSCS storage" on page 1056
- "Putting a queue manager under MSCS control" on page 1058
- "Removing a queue manager from MSCS control" on page 1064

And then gives some useful hints on using MSCS with WebSphere MQ, and details the WebSphere MQ MSCS support utility programs, in the following sections:

- "Hints and tips on using MSCS" on page 1065
- "WebSphere MQ MSCS support utility programs" on page 1068

Introducing MSCS clusters:

MSCS clusters are groups of two or more computers, connected together and configured in such a way that, if one fails, MSCS performs a *failover*, transferring the state data of applications from the failing computer to another computer in the cluster and re-initiating their operation there.

"Using WebSphere MQ with high availability configurations" on page 1040 contains a comparison between MSCS clusters, multi-instance queue managers, and WebSphere MQ clusters.

In this section and its subordinate topics, the term *cluster*, when used by itself, **always** means an MSCS cluster. This is distinct from a WebSphere MQ cluster described elsewhere in this guide.

A two-machine cluster comprises two computers (for example, A and B) that are jointly connected to a network for client access using a *virtual IP address*. They might also be connected to each other by one or more private networks. A and B share at least one disk for the server applications on each to use. There is also another shared disk, which must be a redundant array of independent disks (*RAID*) Level 1, for the exclusive use of MSCS; this is known as the *quorum* disk. MSCS monitors both computers to check that the hardware and software are running correctly.

In a simple setup such as this, both computers have all the applications installed on them, but only computer A runs with live applications; computer B is just running and waiting. If computer A encounters any one of a range of problems, MSCS shuts down the disrupted application in an orderly manner, transfers its state data to the other computer, and re-initiates the application there. This is known as a *failover*. Applications can be made *cluster-aware* so that they interact fully with MSCS and failover gracefully.

A typical setup for a two-computer cluster is as shown in Figure 171.

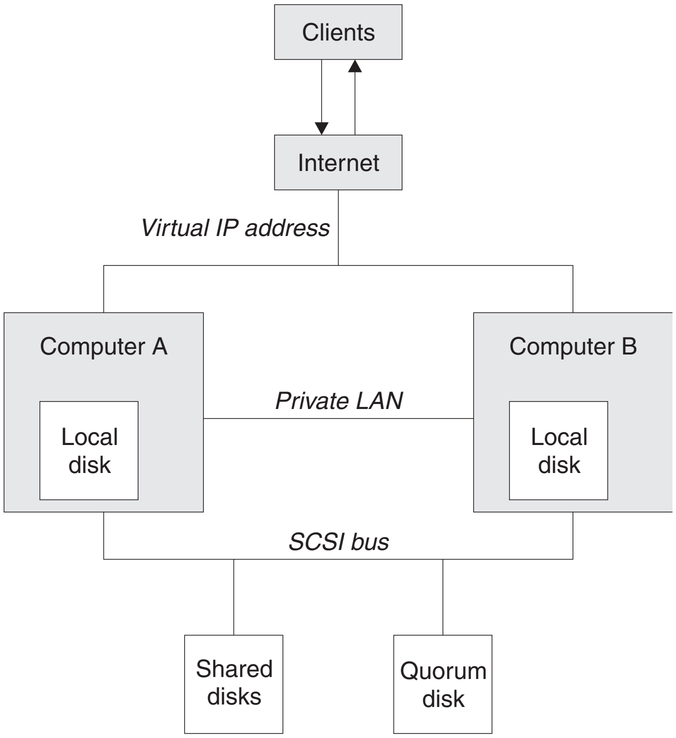


Figure 171. Two-computer MSCS cluster

Each computer can access the shared disk, but only one at a time, under the control of MSCS. In the event of a failover, MSCS switches the access to the other computer. The shared disk itself is usually a RAID, but need not be.

Each computer is connected to the external network for client access, and each has an IP address. However an external client, communicating with this cluster, is aware of only one *virtual IP address*, and MSCS routes the IP traffic within the cluster appropriately.

MSCS also performs its own communications between the two computers, either over one or more private connections or over the public network, for example to monitor their states using the heartbeat, and to synchronize their databases.

Setting up WebSphere MQ for MSCS clustering:

You configure WebSphere MQ for clustering by making the queue manager the unit of failover to MSCS. You define a queue manager as a resource to MSCS, which can then monitor it, and transfer it to another computer in the cluster if there is a problem.

To set your system up for this, you start by installing WebSphere MQ on each computer in the cluster.

As the queue manager is associated with the WebSphere MQ installation name, the WebSphere MQ installation name on all the computers in the cluster should be the same. See Installing and uninstalling.

The queue managers themselves need to exist only on the computer on which you create them. In the event of a failover, the MSCS initiates the queue managers on the other computer. The queue managers, however, must have their log and data files on a cluster shared disk, and not on a local drive. If you have a queue manager already installed on a local drive, you can migrate it using a tool provided with WebSphere MQ; see "Moving a queue manager to MSCS storage" on page 1056. If you want to create new queue managers for use with MSCS, see "Creating a queue manager for use with MSCS" on page 1056.

After installation and migration, use the MSCS Cluster Administrator to make MSCS aware of your queue managers; see "Putting a queue manager under MSCS control" on page 1058.

If you decide to remove a queue manager from MSCS control, use the procedure described in "Removing a queue manager from MSCS control" on page 1064.

Setup symmetry:

When an application switches from one node to the other it must behave in the same way, regardless of node. The best way of ensuring this is to make the environments identical.

If you can, set up a cluster with identical hardware, operating system software, product software, and configuration on each computer. In particular, ensure that all the required software installed on the two computers is identical in terms of version, maintenance level, SupportPacs, paths and exits, and that there is a common namespace (security environment) as described in "MSCS security."

MSCS security:

For successful MSCS security, follow these guidelines.

The guidelines are as follows:

- Make sure you that you have identical software installations on each computer in the cluster.
- Create a common namespace (security environment) across the cluster.
- Make the nodes of the MSCS cluster members of a domain, within which the user account that is the *cluster owner* is a domain account.
- Make the other user accounts on the cluster also domain accounts, so that they are available on both nodes. This is automatically the case if you already have a domain, and the accounts relevant to WebSphere MQ are domain accounts. If you do not currently have a domain, consider setting up a *mini-domain* to cater for the cluster nodes and relevant accounts. Your aim is to make your cluster of two computers look like a single computing resource.

  Remember that an account that is local to one computer does not exist on the other one. Even if you create an account with the same name on the other computer, its security identifier (SID) is different, so, when your application is moved to the other node, the permissions do not exist on that node.

During a failover or move, WebSphere MQ MSCS support ensures that all files that contain queue manager objects have equivalent permissions on the destination node. Explicitly, the code checks that the Administrators and mqm groups, and the SYSTEM account, have full control, and that if `Everyone` had read access on the old node, that permission is added on the destination node.

You can use a domain account to run your WebSphere MQ Service. Make sure that it exists in the local mqm group on each computer in the cluster.

Using multiple queue managers with MSCS:

If you are running more than one queue manager on a computer, you can choose one of these setups.

The setups are as follows:

- All the queue managers in a single group. In this configuration, if a problem occurs with any queue manager, all the queue managers in the group failover to the other computer as a group.
- A single queue manager in each group. In this configuration, if a problem occurs with the queue manager, it alone fails over to the other computer without affecting the other queue managers.
- A mixture of the first two setups.

Cluster modes:

There are two modes in which you might run a cluster system with WebSphere MQ: Active/Passive or Active/Active.

Note: If you are using MSCS together with the Microsoft Transaction Server (COM+), you cannot use Active/Active mode.

Active/Passive mode

In Active/Passive mode, computer A has the running application on it, and computer B is backup, only being used when MSCS detects a problem.

You can use this mode with only one shared disk, but, if any application causes a failover, **all** the applications must be transferred as a group (because only one computer can access the shared disk at a time).

You can configure MSCS with A as the *preferred* computer. Then, when computer A has been repaired or replaced and is working properly again, MSCS detects this and automatically switches the application back to computer A.

If you run more than one queue manager, consider having a separate shared disk for each. Then put each queue manager in a separate group in MSCS. In this way, any queue manager can failover to the other computer without affecting the other queue managers.

Active/Active mode

In Active/Active mode, computers A and B both have running applications, and the groups on each computer are set to use the other computer as backup. If a failure is detected on computer A, MSCS transfers the state data to computer B, and reinitiates the application there. computer B then runs its own application and A's.

For this setup you need at least two shared disks. You can configure MSCS with A as the preferred computer for A's applications, and B as the preferred computer for B's applications. After failover and repair, each application automatically ends up back on its own computer.

For WebSphere MQ this means that you could, for example, run two queue managers, one on each of A and B, with each exploiting the full power of its own computer. After a failure on computer A, both queue managers will run on computer B. This will mean sharing the power of the one computer, with a reduced ability to process large quantities of data at speed. However, your critical applications will still be available while you find and repair the fault on A.

Creating a queue manager for use with MSCS:

This procedure ensures that a new queue manager is created in such a way that it is suitable for preparing and placing under MSCS control.

You start by creating the queue manager with all its resources on a local drive, and then migrate the log files and data files to a shared disk. (You can reverse this operation.) Do **not** attempt to create a queue manager with its resources on a shared drive.

You can create a queue manager for use with MSCS in two ways, either from a command prompt, or in the WebSphere MQ Explorer. The advantage of using a command prompt is that the queue manager is created *stopped* and set to *manual startup*, which is ready for MSCS. (The WebSphere MQ Explorer automatically starts a new queue manager and sets it to automatic startup after creation. You have to change this.)

Creating a queue manager from a command prompt

Follow these steps to create a queue manager from a command prompt, for use with MSCS:

1. Ensure that you have the environment variable `MQSPREFIX` set to refer to a local drive, for example `C:\WebSphere MQ`. If you change this, reboot the machine so that the System account picks up the change. If you do not set the variable, the queue manager is created in the WebSphere MQ default directory for queue managers.
2. Create the queue manager using the **crtmqm** command. For example, to create a queue manager called `mscs_test` in the default directory, use:
   ```
   crtmqm mscs_test
   ```
3. Proceed to “Moving a queue manager to MSCS storage.”

Creating a queue manager using the WebSphere MQ Explorer

Follow these steps to create a queue manager using the WebSphere MQ Explorer, for use with MSCS:

1. Start the WebSphere MQ Explorer from the Start menu.
2. In the Navigator View, expand the tree nodes to find the **Queue Managers** tree node.
3. Right-click the **Queue Managers** tree node, and select **New**->**Queue Manager**. The Create Queue Manager panel is displayed.
4. Complete the dialog (Step 1), then click **Next>**.
5. Complete the dialog (Step 2), then click **Next>**.
6. Complete the dialog (Step 3), ensuring that **Start Queue Manager** and **Create Server Connection Channel** are not selected, then click **Next>**.
7. Complete the dialog (Step 4), then click **Finish**.
8. Proceed to “Moving a queue manager to MSCS storage.”

Moving a queue manager to MSCS storage:

This procedure configures an existing queue manager to make it suitable for putting under MSCS control.

To achieve this, you move the log files and data files to shared disks to make them available to the other computer in the event of a failure. For example, the existing queue manager might have paths such as `C:\WebSphere MQ\log\<QMname>` and `C:\WebSphere MQ\qmgrs\<QMname>`. Do ***not*** try to move the files by hand; use the utility program supplied as part of WebSphere MQ MSCS Support as described in this topic.

If the queue manager being moved uses SSL connections and the SSL key repository is in the queue manager data directory on the local machine, then the key repository will be moved with the rest of the queue manager to the shared disk. By default, the queue manager attribute that specifies the SSL key repository location, SSLKEYR, is set to *MQ_INSTALLATION_PATH*\qmgrs\QMGRNAME\ssl\key, which is under the queue manager data directory. *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed. The hamvmqm command does not modify this queue manager attribute. In this situation you must modify the queue manager attribute, SSLKEYR, using the WebSphere MQ Explorer or the MQSC command ALTER QMGR, to point to the new SSL key repository file.

The procedure is as follows:

1. Shut down the queue manager, and check that there are no errors.
2. If the queue manager's log files or queue files are already stored on a shared disk, skip the rest of this procedure and proceed directly to “Putting a queue manager under MSCS control” on page 1058.
3. Make a full media backup of the queue files and log files and store the backup in a safe place (see “Queue manager log files” on page 1066 for why this is important).
4. If you already have a suitable shared disk resource proceed to step 6. Otherwise, using the MSCS Cluster Administrator to create a resource of type *shared disk* with sufficient capacity to store the queue manager log files and data (queue) files.
5. Test the shared disk by using the MSCS Cluster Administrator to move it from one cluster node to the other and back again.
6. Make sure that the shared disk is online on the cluster node where the queue manager log and data files are stored locally.
7. Run the utility program to move the queue manager as follows:

   ```
   hamvmqm /m qmname /dd "e:\WebSphere MQ" /ld "e:\WebSphere MQ\log"
   ```

   substituting your queue manager name for *qmname*, your shared disk drive letter for *e*, and your chosen directory for *WebSphere MQ*. The directories are created if they do not already exist.
8. Test the queue manager to ensure that it works, using the WebSphere MQ Explorer. For example:
   a. Right-click the queue manager tree node, then select **Start**. The queue manager starts.
   b. Right-click the **Queues** tree node, then select **New**->**Local Queue...**, and give the queue a name.
   c. Click **Finish**.
   d. Right-click the queue, then select **Put Test Message...**. The Put Test Message panel is displayed.
   e. Type some message text, then click **Put Test Message**, and close the panel.
   f. Right-click the queue, then select **Browse Messages...**. The Message Browser panel is displayed.
   g. Ensure your message is on the queue, then click **Close** . The Message Browser panel closes.
   h. Right-click the queue, then select **Clear Messages...**. The messages on the queue are cleared.
   i. Right-click the queue, then select **Delete...**. A confirmation panel is displayed, click **OK**. The queue is deleted.
   j. Right-click the queue manager tree node, then select **Stop...**. The End Queue Manager panel is displayed.
   k. Click **OK**. The queue manager stops.
9. As WebSphere MQ Administrator ensure that the startup attribute of the queue manager is set to manual. In the WebSphere MQ Explorer, set the Startup field to manual in the queue manager properties panel.
10. Proceed to “Putting a queue manager under MSCS control” on page 1058.

Putting a queue manager under MSCS control:

The tasks involved in placing a queue manager under MSCS control, including prerequisite tasks.

Before you put a queue manager under MSCS control

Before you put a queue manager under MSCS control, perform the following tasks:

1. Ensure that WebSphere MQ and its MSCS Support are installed on both machines in the cluster and that the software on each computer is identical, as described in “Setting up WebSphere MQ for MSCS clustering” on page 1053.
2. Use the **haregtyp** utility program to register WebSphere MQ as an MSCS resource type on all the cluster nodes. See “WebSphere MQ MSCS support utility programs” on page 1068 for additional information.
3. If you have not yet created the queue manager, see “Creating a queue manager for use with MSCS” on page 1056.
4. If you have created the queue manager, or it already exists, ensure that you have carried out the procedure in “Moving a queue manager to MSCS storage” on page 1056.
5. Stop the queue manager, if it is running, using either a command prompt or the WebSphere MQ Explorer.
6. Test MSCS operation of the shared drives before going on to either of the following Windows procedures in this topic.

Windows Server 2012

Attention: MSCS support is delivered in WebSphere MQ 7.5 using a 32-bit DLL. Due to a restriction in Windows 2012, the WebSphere MQ queue manager does not fail over following a restart.

Microsoft have deprecated the use of 32-bit DLLs with Windows 2012 and, therefore, no operating system fix is currently available for this issue. IBM does not provide a 64-bit library for WebSphere MQ 7.5.

From IBM MQ 8.0 a 64-bit library is available, so you must use this version of the product for full MSCS functionality with Windows 2012, and later.

To place a queue manager under MSCS control on Windows Server 2012, use the following procedure:

1. Log in to the cluster node computer hosting the queue manager, or log in to a remote workstation as a user with cluster administration permissions, and connect to the cluster node hosting the queue manager.
2. Start the Failover Cluster Management tool.
3. Right-click **Failover Cluster Management > Connect Cluster ...** to open a connection to the cluster.
4. In contrast to the group scheme used in the MSCS Cluster Administrator on previous versions of Windows, the Failover Cluster Management tool uses the concept of services and applications. A configured service or application contains all the resources necessary for one application to be clustered. You can configure a queue manager under MSCS as follows:
   a. Right-click on the cluster and select **Configure Role** to start the configuration wizard.
   b. Select **Other Server** on the "Select Service or Application" panel.
   c. Select an appropriate IP address as a client access point.

      This address should be an unused IP address to be used by clients and other queue managers to connect to the *virtual* queue manager. This IP address is not the normal (static) address of either node; it is an additional address that *floats* between them. Although MSCS handles the routing of this address, it does **not** verify that the address can be reached.
   d. Assign a storage device for exclusive use by the queue manager. This device needs to be created as a resource instance before it can be assigned.

You can use one drive to store both the logs and queue files, or you can split them up across drives. In either case, if each queue manager has its own shared disk, ensure that all drives used by this queue manager are exclusive to this queue manager, that is, that nothing else relies on the drives. Also ensure that you create a resource instance for every drive that the queue manager uses.

The resource type for a drive depends on the SCSI support you are using; refer to your SCSI adapter instructions. There might already be groups and resources for each of the shared drives. If so, you do not need to create the resource instance for each drive. Move it from its current group to the one created for the queue manager.

For each drive resource, set possible owners to both nodes. Set dependent resources to none.

e. Select the **IBM MQSeries MSCS** resource on the "Select Resource Type" panel.

f. Complete the remaining steps in the wizard.

5. Before bringing the resource online, the IBM MQSeries MSCS resource needs additional configuration:

   a. Select the newly defined service which contains a resource called 'New IBM MQSeries MSCS'.

   b. Right-click **Properties** on the MQ resource.

   c. Configure the resource:

      - `Name`; choose a name that makes it easy to identify which queue manager it is for.
      - `Run in a separate Resource Monitor`; for better isolation
      - `Possible owners`; set both nodes
      - `Dependencies`; add the drive and IP address for this queue manager.

        Warning: Failure to add these dependencies means that WebSphere MQ attempts to write the queue manager status to the wrong cluster disk during failovers. Because many processes might be attempting to write to this disk simultaneously, some WebSphere MQ processes could be blocked from running.
      - `Parameters`; as follows:
        - `QueueManagerName` (required); the name of the queue manager that this resource is to control. This queue manager must exist on the local computer.
        - `PostOnlineCommand` (optional); you can specify a program to run whenever the queue manager resource changes its state from offline to online. For more details see "PostOnlineCommand and PreOfflineCommand" on page 1067.
        - `PreOfflineCommand` (optional); you can specify a program to run whenever the queue manager resource changes its state from online to offline. For more details see "PostOnlineCommand and PreOfflineCommand" on page 1067.

          Note: The *looksAlive* poll interval is set to default value of 5000 ms. The *isAlive* poll interval is set to default value of 60000 ms. These defaults can only be modified after the resource definition has been completed. For further details see "Summary of looksAlive and isAlive polling" on page 1063.

   d. Optionally, set a preferred node (but note the comments in "Using preferred nodes" on page 1067)

   e. The *Failover Policy* is set by default to sensible values, but you can tune the thresholds and periods that control *Resource Failover* and *Group Failover* to match the loads placed on the queue manager.

6. Test the queue manager by bringing it online in the MSCS Cluster Administrator and subjecting it to a test workload. If you are experimenting with a test queue manager, use the WebSphere MQ Explorer. For example:

   a. Right-click the **Queues** tree node, then select **New**->**Local Queue...**, and give the queue a name.

   b. Click **Finish**. The queue is created, and displayed in the content view.

   c. Right-click the queue, then select **Put Test Message...**. The Put Test Message panel is displayed.

   d. Type some message text, then click **Put Test Message**, and close the panel.

   e. Right-click the queue, then select **Browse Messages...**. The Message Browser panel is displayed.

- f. Ensure that your message is on the queue, then click **Close** . The Message Browser panel closes.
- g. Right-click the queue, then select **Clear Messages...**. The messages on the queue are cleared.
- h. Right-click the queue, then select **Delete...**. A confirmation panel is displayed, click **OK**. The queue is deleted.

7. Test that the queue manager can be taken offline and back online using the MSCS Cluster Administrator.
8. Simulate a failover.

   In the MSCS Cluster Administrator, right-click the group containing the queue manager and select `Move Group`. This can take some minutes to do. (If at other times you want to move a queue manager to another node quickly, follow the procedure in “Moving a queue manager to MSCS storage” on page 1056.) You can also right-click and select `Initiate Failure`; the action (local restart or failover) depends on the current state and the configuration settings.

Windows Server 2008

To place a queue manager under MSCS control on Windows Server 2008, use the following procedure:

1. Log in to the cluster node computer hosting the queue manager, or log in to a remote workstation as a user with cluster administration permissions, and connect to the cluster node hosting the queue manager.
2. Start the Failover Cluster Management tool.
3. Right-click **Failover Cluster Management > Manage a Cluster ...** to open a connection to the cluster.
4. In contrast to the group scheme used in the MSCS Cluster Administrator on previous versions of Windows, the Failover Cluster Management tool uses the concept of services and applications. A configured service or application contains all the resources necessary for one application to be clustered. You can configure a queue manager under MSCS as follows:
   - a. Right-click **Services and Applications > Configure a Service or Application ...** to start the configuration wizard.
   - b. Select **Other Server** on the "Select Service or Application" panel.
   - c. Select an appropriate IP address as a client access point.

     This address should be an unused IP address to be used by clients and other queue managers to connect to the *virtual* queue manager. This IP address is not the normal (static) address of either node; it is an additional address that *floats* between them. Although MSCS handles the routing of this address, it does **not** verify that the address can be reached.
   - d. Assign a storage device for exclusive use by the queue manager. This device needs to be created as a resource instance before it can be assigned.

     You can use one drive to store both the logs and queue files, or you can split them up across drives. In either case, if each queue manager has its own shared disk, ensure that all drives used by this queue manager are exclusive to this queue manager, that is, that nothing else relies on the drives. Also ensure that you create a resource instance for every drive that the queue manager uses.

     The resource type for a drive depends on the SCSI support you are using; refer to your SCSI adapter instructions. There might already be groups and resources for each of the shared drives. If so, you do not need to create the resource instance for each drive. Move it from its current group to the one created for the queue manager.

     For each drive resource, set possible owners to both nodes. Set dependent resources to none.
   - e. Select the **IBM MQSeries MSCS** resource on the "Select Resource Type" panel.
   - f. Complete the remaining steps in the wizard.
5. Before bringing the resource online, the IBM MQSeries MSCS resource needs additional configuration:
   - a. Select the newly defined service which contains a resource called 'New IBM MQSeries MSCS'.
   - b. Right-click **Properties** on the MQ resource.

c. Configure the resource:
   - `Name`; choose a name that makes it easy to identify which queue manager it is for.
   - `Run in a separate Resource Monitor`; for better isolation
   - `Possible owners`; set both nodes
   - `Dependencies`; add the drive and IP address for this queue manager.

     Warning: Failure to add these dependencies means that WebSphere MQ attempts to write the queue manager status to the wrong cluster disk during failovers. Because many processes might be attempting to write to this disk simultaneously, some WebSphere MQ processes could be blocked from running.
   - `Parameters`; as follows:
     - `QueueManagerName` (required); the name of the queue manager that this resource is to control. This queue manager must exist on the local computer.
     - `PostOnlineCommand` (optional); you can specify a program to run whenever the queue manager resource changes its state from offline to online. For more details see "PostOnlineCommand and PreOfflineCommand" on page 1067.
     - `PreOfflineCommand` (optional); you can specify a program to run whenever the queue manager resource changes its state from online to offline. For more details see "PostOnlineCommand and PreOfflineCommand" on page 1067.

       Note: The *looksAlive* poll interval is set to default value of 5000 ms. The *isAlive* poll interval is set to default value of 60000 ms. These defaults can only be modified after the resource definition has been completed. For further details see "Summary of looksAlive and isAlive polling" on page 1063.

d. Optionally, set a preferred node (but note the comments in "Using preferred nodes" on page 1067)

e. The *Failover Policy* is set by default to sensible values, but you can tune the thresholds and periods that control *Resource Failover* and *Group Failover* to match the loads placed on the queue manager.

6. Test the queue manager by bringing it online in the MSCS Cluster Administrator and subjecting it to a test workload. If you are experimenting with a test queue manager, use the WebSphere MQ Explorer. For example:
   a. Right-click the **Queues** tree node, then select **New**->**Local Queue...**, and give the queue a name.
   b. Click **Finish**. The queue is created, and displayed in the content view.
   c. Right-click the queue, then select **Put Test Message...**. The Put Test Message panel is displayed.
   d. Type some message text, then click **Put Test Message**, and close the panel.
   e. Right-click the queue, then select **Browse Messages...**. The Message Browser panel is displayed.
   f. Ensure that your message is on the queue, then click **Close** . The Message Browser panel closes.
   g. Right-click the queue, then select **Clear Messages...**. The messages on the queue are cleared.
   h. Right-click the queue, then select **Delete...**. A confirmation panel is displayed, click **OK**. The queue is deleted.
7. Test that the queue manager can be taken offline and back online using the MSCS Cluster Administrator.
8. Simulate a failover.

   In the MSCS Cluster Administrator, right-click the group containing the queue manager and select `Move Group`. This can take some minutes to do. (If at other times you want to move a queue manager to another node quickly, follow the procedure in "Moving a queue manager to MSCS storage" on page 1056.) You can also right-click and select `Initiate Failure`; the action (local restart or failover) depends on the current state and the configuration settings.

Windows 2003

To place a queue manager under MSCS control on Windows 2003, use the following procedure:

1. Log in to the cluster node computer hosting the queue manager, or log in to a remote workstation as a user with cluster administration permissions, and connect to the cluster node hosting the queue manager.
2. Start the MSCS Cluster Administrator.
3. Open a connection to the cluster.
4. Create an MSCS group to be used to contain the resources for the queue manager. Name the group in such a way that it is obvious which queue manager it relates to. Each group can contain multiple queue managers, as described in "Using multiple queue managers with MSCS" on page 1055.

   Use the group for all the remaining steps.
5. Create a resource instance for each of the SCSI logical drives that the queue manager uses.

   You can use one drive to store both the logs and queue files, or you can split them up across drives. In either case, if each queue manager has its own shared disk, ensure that all drives used by this queue manager are exclusive to this queue manager, that is, that nothing else relies on the drives. Also ensure that you create a resource instance for every drive that the queue manager uses.

   The resource type for a drive depends on the SCSI support you are using; refer to your SCSI adapter instructions. There might already be groups and resources for each of the shared drives. If so, you do not need to create the resource instance for each drive. Move it from its current group to the one created for the queue manager.

   For each drive resource, set possible owners to both nodes. Set dependent resources to none.
6. Create a resource instance for the IP address.

   Create an IP address resource (resource type *IP address*). This address should be an unused IP address to be used by clients and other queue managers to connect to the *virtual* queue manager. This IP address is not the normal (static) address of either node; it is an additional address that *floats* between them. Although MSCS handles the routing of this address, it does **not** verify that the address can be reached.
7. Create a resource instance for the queue manager.

   Create a resource of type *IBM WebSphere MQ MSCS*. The wizard prompts you for various items, including the following:

   - `Name`; choose a name that makes it easy to identify which queue manager it is for.
   - `Add to group`; use the group that you created
   - `Run in a separate Resource Monitor`; for better isolation
   - `Possible owners`; set both nodes
   - `Dependencies`; add the drive and IP address for this queue manager.

     Warning: Failure to add these dependencies means that WebSphere MQ attempts to write the queue manager status to the wrong cluster disk during failovers. Because many processes might be attempting to write to this disk simultaneously, some WebSphere MQ processes could be blocked from running.
   - `Parameters`; as follows:
     - `QueueManagerName` (required); the name of the queue manager that this resource is to control. This queue manager must exist on the local computer.
     - `PostOnlineCommand` (optional); you can specify a program to run whenever the queue manager resource changes its state from offline to online. For more details see "PostOnlineCommand and PreOfflineCommand" on page 1067.
     - `PreOfflineCommand` (optional); you can specify a program to run whenever the queue manager resource changes its state from online to offline. For more details see "PostOnlineCommand and PreOfflineCommand" on page 1067.

Note: The *looksAlive* poll interval is set to default value of 5000 ms. The *isAlive* poll interval is set to default value of 30000 ms. These defaults can only be modified after the resource definition has been completed. For further details see "Summary of looksAlive and isAlive polling."

8. Optionally, set a preferred node (but note the comments in "Using preferred nodes" on page 1067)
9. The *Failover Policy* (as defined in the properties for the group) is set by default to sensible values, but you can tune the thresholds and periods that control *Resource Failover* and *Group Failover* to match the loads placed on the queue manager.
10. Test the queue manager by bringing it online in the MSCS Cluster Administrator and subjecting it to a test workload. If you are experimenting with a test queue manager, use the WebSphere MQ Explorer. For example:
    a. Right-click the **Queues** tree node, then select **New**->**Local Queue...**, and give the queue a name.
    b. Click **Finish**. The queue is created, and displayed in the content view.
    c. Right-click the queue, then select **Put Test Message...**. The Put Test Message panel is displayed.
    d. Type some message text, then click **Put Test Message**, and close the panel.
    e. Right-click the queue, then select **Browse Messages...**. The Message Browser panel is displayed.
    f. Ensure that your message is on the queue, then click **Close** . The Message Browser panel closes.
    g. Right-click the queue, then select **Clear Messages...**. The messages on the queue are cleared.
    h. Right-click the queue, then select **Delete...**. A confirmation panel is displayed, click **OK**. The queue is deleted.
11. Test that the queue manager can be taken offline and back online using the MSCS Cluster Administrator.
12. Simulate a failover.

    In the MSCS Cluster Administrator, right-click the group containing the queue manager and select `Move Group`. This can take some minutes to do. (If at other times you want to move a queue manager to another node quickly, follow the procedure in "Moving a queue manager to MSCS storage" on page 1056.) You can also right-click and select `Initiate Failure`; the action (local restart or failover) depends on the current state and the configuration settings.

Summary of looksAlive and isAlive polling:

looksAlive and *isAlive* are intervals at which MSCS calls back into the resource types supplied library code and requests that the resource performs checks to determine the working status of itself. This ultimately determines if MSCS attempts to fail over the resource.

On every occasion that the *looksAlive* interval elapses (default 5000 ms), the queue manager resource is called to perform its own check to determine if its status is satisfactory.

On every occasion that the *isAlive* interval elapses (default 30000 ms), another call is made to the queue manager resource for it to perform another check to determine if the resource is functioning correctly. This enables two levels of resource type checking.

1. A *looksAlive* status check to establish if the resource appears to be functioning.
2. A more significant *isAlive* check that determines if the queue manager resource is active.

If the queue manager resource is determined not to be active, MSCS, based on other advanced MSCS options, triggers a fail over for the resource and associated dependant resources to another node in the cluster. For further information, see MSCS documentation.

Removing a queue manager from MSCS control:

You can remove queue managers from MSCS control, and return them to manual administration.

You do not need to remove queue managers from MSCS control for maintenance operations. You can do that by taking a queue manager offline temporarily, using the MSCS Cluster Administrator. Removing a queue manager from MSCS control is a more permanent change; only do it if you decide that you no longer want MSCS to have any further control of the queue manager.

If the queue manager being removed uses SSL connections you must modify the queue manager attribute, SSLKEYR, using the WebSphere MQ Explorer or the MQSC command `ALTER QMGR`, to point to the SSL key repository file on the local directory.

The procedure is:

1. Take the queue manager resource offline using the MSCS Cluster Administrator, as described in "Taking a queue manager offline from MSCS"
2. Destroy the resource instance. This does not destroy the queue manager.
3. Optionally, migrate the queue manager files back from shared drives to local drives. To do this, see "Returning a queue manager from MSCS storage."
4. Test the queue manager.

Taking a queue manager offline from MSCS

To take a queue manager offline from MSCS, perform the following steps:

1. Start the MSCS Cluster Administrator.
2. Open a connection to the cluster.
3. Select `Groups`, and open the group containing the queue manager to be moved.
4. Select the queue manager resource.
5. Right-click it and select `Offline`.
6. Wait for completion.

Returning a queue manager from MSCS storage

This procedure configures the queue manager to be back on its computer's local drive, that is, it becomes a *normal* WebSphere MQ queue manager. To achieve this, you move the log files and data files from the shared disks. For example, the existing queue manager might have paths such as `E:\WebSphere MQ\log\<QMname>` and `E:\WebSphere MQ\qmgrs\<QMname>`. Do not try to move the files by hand; use the **hamvmqm** utility program supplied as part of WebSphere MQ MSCS Support:

1. Shut down the queue manager, and check that there are no errors.
2. Make a full media backup of the queue files and log files and store the backup in a safe place (see "Queue manager log files" on page 1066 for why this is important).
3. Decide which local drive to use and ensure that it has sufficient capacity to store the queue manager log files and data (queue) files.
4. Make sure that the shared disk on which the files currently reside is online on the cluster node to which to move the queue manager log and data files.
5. Run the utility program to move the queue manager as follows:

   ```
   hamvmqm /m qmname /dd "c:\WebSphere MQ" /ld "c:\WebSphere MQ\log"
   ```

   substituting your queue manager name for *qmname*, your local disk drive letter for *c*, and your chosen directory for *WebSphere MQ* (the directories are created if they do not already exist).
6. Test the queue manager to ensure that it works (as described in "Moving a queue manager to MSCS storage" on page 1056).

Hints and tips on using MSCS:

This section contains some general information to help you use WebSphere MQ support for MSCS effectively.

This section contains some general information to help you use WebSphere MQ support for MSCS effectively.

How long does it take to fail a queue manager over from one machine to the other? This depends heavily on the amount of workload on the queue manager and on the mix of traffic, for example, how much of it is persistent, within sync point, and how much committed before the failure. IBM tests have given failover and failback times of about a minute. This was on a very lightly loaded queue manager and actual times will vary considerably depending on load.

Verifying that MSCS is working:

Follow these steps to ensure that you have a running MSCS cluster.

The task descriptions starting with "Creating a queue manager for use with MSCS" on page 1056 assume that you have a running MSCS cluster within which you can create, migrate, and destroy resources. If you want to make sure that you have such a cluster:

1. Using the MSCS Cluster Administrator, create a group.
2. Within that group, create an instance of a generic application resource, specifying the system clock (path name `C:\winnt\system32\clock.exe` and working directory of `C:\`).
3. Make sure that you can bring the resource online, that you can move the group that contains it to the other node, and that you can take the resource offline.

Manual startup:

For a queue manager managed by MSCS, you ***must*** set the startup attribute to manual. This ensures that the WebSphere MQ MSCS support can restart the IBM MQSeries Service without immediately starting the queue manager.

The WebSphere MQ MSCS support needs to be able to restart the service so that it can perform monitoring and control, but must itself remain in control of which queue managers are running, and on which machines. See "Moving a queue manager to MSCS storage" on page 1056 for more information.

MSCS and queue managers:

Considerations concerning queue managers when using MSCS.

Creating a matching queue manager on the other node

For clustering to work with WebSphere MQ, you need an identical queue manager on node B for each one on node A. However, you do not need to explicitly create the second one. You can create or prepare a queue manager on one node, move it to the other node as described in "Moving a queue manager to MSCS storage" on page 1056, and it is fully duplicated on that node.

Default queue managers

Do not use a default queue manager under MSCS control. A queue manager does not have a property that makes it the default; WebSphere MQ keeps its own separate record. If you move a queue manager set to be the default to the other computer on failover, it does not become the default there. Make all your applications refer to specific queue managers by name.

Deleting a queue manager

Once a queue manager has moved node, its details exist in the registry on both computers. When you want to delete it, do so as normal on one computer, and then run the utility described in "WebSphere MQ MSCS support utility programs" on page 1068 to clean up the registry on the other computer.

Support for existing queue managers

You can put an existing queue manager under MSCS control, provided that you can put your queue manager log files and queue files on a disk that is on the shared SCSI bus between the two machines (see Figure 171 on page 1053). You need to take the queue manager offline briefly while the MSCS Resource is created.

If you want to create a new queue manager, create it independently of MSCS, test it, then put it under MSCS control. See:

- "Creating a queue manager for use with MSCS" on page 1056
- "Moving a queue manager to MSCS storage" on page 1056
- "Putting a queue manager under MSCS control" on page 1058

Telling MSCS which queue managers to manage

You choose which queue managers are placed under MSCS control by using the MSCS Cluster Administrator to create a resource instance for each such queue manager. This process presents you with a list of resources from which to select the queue manager that you want that instance to manage.

Queue manager log files

When you move a queue manager to MSCS storage, you move its log and data files to a shared disk (for an example see "Moving a queue manager to MSCS storage" on page 1056).

It is advisable before you move, to shut the queue manager cleanly and take a full backup of the data files and log files.

Multiple queue managers

WebSphere MQ MSCS support allows you to run multiple queue managers on each machine and to place individual queue managers under MSCS control.

Always use MSCS to manage clusters:

Do not try to perform start and stop operations directly on any queue manager under the control of MSCS, using either the control commands or the WebSphere MQ Explorer. Instead, use MSCS Cluster Administrator to bring the queue manager online or take it offline.

Using the MSCS Cluster Administrator is partly to prevent possible confusion caused by MSCS reporting that the queue manager is offline, when in fact you have started it outside the control of MSCS. More seriously, stopping a queue manager without using MSCS is detected by MSCS as a failure, initiating failover to the other node.

Working in Active/Active mode:

Both computers in the MSCS cluster can run queue managers in Active/Active mode. You do not need to have a completely idle machine acting as standby (but you can, if you want, in Active/Passive Mode).

If you plan to use both machines to run workload, provide each with sufficient capacity (processor, memory, secondary storage) to run the entire cluster workload at a satisfactory level of performance.

Note: If you are using MSCS together with Microsoft Transaction Server (COM+), you **cannot** use Active/Active mode. This is because, to use WebSphere MQ with MSCS and COM+:

- Application components that use WebSphere MQ's COM+ support must run on the same computer as the Distributed Transaction Coordinator (DTC), a part of COM+.
- The queue manager must also run on the same computer.
- The DTC must be configured as an MSCS resource, and can therefore run on only one of the computers in the cluster at any time.

PostOnlineCommand and PreOfflineCommand:

Use these commands to integrate WebSphere MQ MSCS support with other systems. You can use them to issue WebSphere MQ commands, wih some restrictions.

Specify these commands in the Parameters to a resource of type `IBM WebSphere MQ MSCS`. You can use them to integrate WebSphere MQ MSCS support with other systems or procedures. For example, you could specify the name of a program that sends a mail message, activates a pager, or generates some other form of alert to be captured by another monitoring system.

PostOnlineCommand is invoked when the resource changes from offline to online; PreOfflineCommand is invoked for a change from online to offline. When invoked these commands are run, by default, from the Windows system directory. Because WebSphere MQ uses a 32-bit resource monitor process, on Windows 64-bit systems, this is the `\Windows\SysWOW64` directory rather than `\Windows\system32`. For more information, see the Microsoft documentation about file redirection in a Windows x64 environment. Both commands run under the user account used to run the MSCS Cluster Service; and are invoked asynchronously; WebSphere MQ MSCS support does not wait for them to complete before continuing. This eliminates any risk that they might block or delay further cluster operations.

You can also use these commands to issue WebSphere MQ commands, for example to restart Requester channels. However, the commands are run at the point in time when the queue manager's state changes so they are not intended to perform long-running functions and must not make assumptions about the current state of the queue manager; it is quite possible that, immediately after the queue manager was brought online, an administrator issued an offline command.

If you want to run programs that depend on the state of the queue manager, consider creating instances of the `MSCS Generic Application` resource type, placing them in the same MSCS group as the queue manager resource, and making them dependent on the queue manager resource.

Using preferred nodes:

It can be useful when using Active/Active mode to configure a *preferred node* for each queue manager. However, in general it is better not to set a preferred node but to rely on a manual failback.

Unlike some other relatively stateless resources, a queue manager can take a while to fail over (or back) from one node to the other. To avoid unnecessary outages, test the recovered node before failing a queue manager back to it. This precludes use of the `immediate` failback setting. You can configure failback to occur between certain times of day.

Probably the safest route is to move the queue manager back manually to the required node, when you are certain that the node is fully recovered. This precludes use of the `preferred node` option.

If COM+ errors occur in the Application Event log:

When you install WebSphere MQ on a newly-installed MSCS cluster, you might find an error with Source COM+ and Event ID 4691 reported in the Application Event log.

This means that you are trying to run WebSphere MQ on a Microsoft Cluster Server (MSCS) environment when the Microsoft Distributed Transaction Coordinator (MSDTC) has not been configured to run in such an environment. For information on configuring MSDTC in a clustered environment, refer to Microsoft documentation.

WebSphere MQ MSCS support utility programs:

A list of the WebSphere MQ support for MSCS utility programs that you can run at a command prompt.

WebSphere MQ support for MSCS includes the following utility programs:

Register/unregister the resource type

`haregtyp.exe`

After you *unregister* the WebSphere MQ MSCS resource type you can no longer create any resources of that type. MSCS does not let you unregister a resource type if you still have instances of that type within the cluster:

1. Using the MSCS Cluster Administrator, stop any queue managers that are running under MSCS control, by taking them offline as described in “Taking a queue manager offline from MSCS” on page 1064.
2. Using the MSCS Cluster Administrator, delete the resource instances.
3. At a command prompt, unregister the resource type by entering the following command:

```
haregtyp /u
```

If you want to *register* the type (or re-register it at a later time), enter the following command at a command prompt:

```
haregtyp /r
```

After successfully registering the MSCS libraries, you must reboot the system if you have not done so since installing WebSphere MQ.

Move a queue manager to MSCS storage

`hamvmqm.exe`

See “Moving a queue manager to MSCS storage” on page 1056.

Delete a queue manager from a node

`hadltmqm.exe`

Consider the case where you have had a queue manager in your cluster, it has been moved from one node to another, and now you want to destroy it. Use the WebSphere MQ Explorer to delete it on the node where it currently is. The registry entries for it still exist on the other computer. To delete these, enter the following command at a prompt on that computer:

```
hadltmqm /m qmname
```

where `qmname` is the name of the queue manager to remove.

Check and save setup details

`amqmsysn.exe`

This utility presents a dialog showing full details of your WebSphere MQ MSCS Support setup, such as might be requested if you call IBM support. There is an option to save the details to a file.

Multi-instance queue managers

Multi-instance queue managers are instances of the same queue manager configured on different servers. One instance of the queue manager is defined as the active instance and another instance is defined as the standby instance. If the active instance fails, the multi-instance queue manager restarts automatically on the standby server.

Figure 172 shows a multi-instance configuration for QM1. WebSphere MQ is installed on two servers, one of which is a spare. One queue manager, QM1, has been created. One instance of QM1 is active, and is running on one server. The other instance of QM1 is running in standby on the other server, doing no active processing, but ready to take over from the active instance of QM1, if the active instance fails.

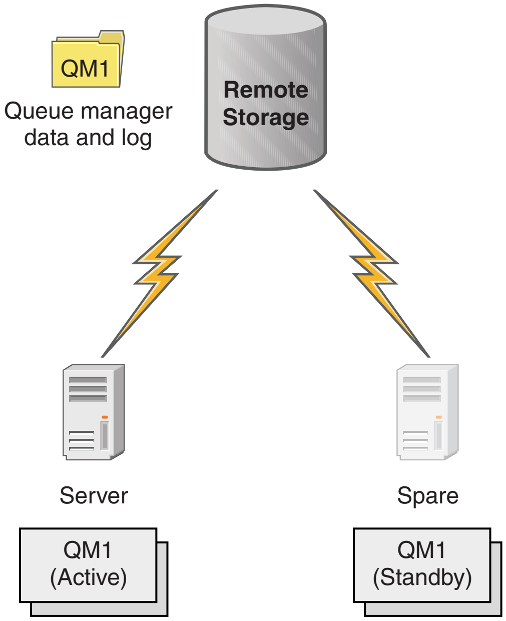


Figure 172. Multi-instance queue manager

When you intend to use a queue manager as a multi-instance queue manager, create a single queue manager on one of the servers using the **crtmqm** command, placing its queue manager data and logs in shared network storage. On the other server, rather than create the queue manager again, use the **addmqinf** command to create a reference to the queue manager data and logs on the network storage.

You can now run the queue manager from either of the servers. Each of the servers references the same queue manager data and logs; there is only one queue manager, and it is active on only one server at a time.

The queue manager can run either as a single instance queue manager, or as a multi-instance queue manager. In both cases only one instance of the queue manager is running, processing requests. The difference is that when running as a multi-instance queue manager, the server that is not running the active instance of the queue manager runs as a standby instance, ready to take over from the active instance automatically if the active server fails.

The only control you have over which instance becomes active first is the order in which you start the queue manager on the two servers. The first instance to acquire read/write locks to the queue manager data becomes the active instance.

You can swap the active instance to the other server, once it has started, by stopping the active instance using the switchover option to transfer control to the standby.

The active instance of QM1 has exclusive access to the shared queue manager data and logs folders when it is running. The standby instance of QM1 detects when the active instance has failed, and becomes the active instance. It takes over the QM1 data and logs in the state they were left by the active instance, and accepts reconnections from clients and channels.

The active instance might fail for various reasons that result in the standby taking over:

- Failure of the server hosting the active queue manager instance.
- Failure of connectivity between the server hosting the active queue manager instance and the file system.
- Unresponsiveness of queue manager processes, detected by WebSphere MQ, which then shuts down the queue manager.

You can add the queue manager configuration information to multiple servers, and choose any two servers to run as the active/standby pair. There is a limit of a total of two instances. You cannot have two standby instances and one active instance.

A multi-instance queue manager is one part of a high availability solution. You need some additional components to build a useful high availability solution.

- Client and channel reconnection to transfer WebSphere MQ connections to the computer that takes over running the active queue manager instance.
- A high performance shared network file system (NFS) that manages locks correctly and provides protection against media and file server failure.

  Important: You must stop all multi-instance queue manager instances that are running in your environment before you can perform maintenance on the NFS drive. Make sure that you have queue manager configuration backups to recover, in the event of an NFS failure.
- Resilient networks and power supplies to eliminate single points of failure in the basic infrastructure.
- Applications that tolerate failover. In particular you need to pay close attention to the behavior of transactional applications, and to applications that browse WebSphere MQ queues.
- Monitoring and management of the active and standby instances to ensure that they are running, and to restart active instances that have failed. Although multi-instance queue managers restart automatically, you need to be sure that your standby instances are running, ready to take over, and that failed instances are brought back online as new standby instances.

WebSphere MQ MQI clients and channels reconnect automatically to the standby queue manager when it becomes active. More information about reconnection, and the other components in a high availability solution can be found in related topics. Automatic client reconnect is not supported by WebSphere MQ classes for Java.

Supported platforms

You can create a multi-instance queue manager on any of the non-z/OS platforms from version 7.0.1.

Automatic client reconnection is supported for MQI clients from version 7.0.1 onwards.

Create a multi-instance queue manager:

Create a multi-instance queue manager, creating the queue manager on one server, and configuring IBM WebSphere MQ on another server. Multi-instance queue managers shared queue manager data and logs.

Most of the effort involved in creating a multi-instance queue manager is the task of setting up the shared queue manager data and log files. You must create shared directories on network storage, and make the directories available to other servers using network shares. These tasks need to be performed by someone with administrative authority, such as *root* on UNIX and Linux systems. The steps are as follows:

1. Create the shares for the data and log files.
2. Create the queue manager on one server.
3. Run the command **dspmqinf** on the first server to collect the queue manager configuration data and copy it into the clipboard.
4. Run the command **addmqinf** with the copied data to create the queue manager configuration on the second server.

You do not run **crtmqm** to create the queue manager again on the second server.

File access control

You need to take care that the user and group mqm on all other servers have permission to access the shares.

On UNIX and Linux, you need to make the uid and gid of mqm the same on all the systems. You might need to edit /etc/passwd on each system to set a common uid and gid for mqm, and then reboot your system.

On Microsoft Windows, the user ID that is running the queue manager processes must have full control permission to the directories containing the queue manager data and log files. You can configure the permission in two ways:

1. Create a queue manager with a global group as the alternative security principal. Authorize the global group to have full control access to the directories containing queue manager data and log files; see “Secure shared queue manager data and log directories and files on Windows” on page 1096. Make the user ID that is running the queue manager a member of the global group. You cannot make a local user a member of a global group, so the queue manager processes must run under a domain user ID. The domain user ID must be a member of the local group mqm. The task, “Create a multi-instance queue manager on domain workstations or servers” on page 1073, demonstrates how to set up a multi-instance queue manager using files secured in this way.
2. Create a queue manager on the domain controller, so that the local mqm group has domain scope, “domain local”. Secure the file share with the domain local mqm, and run queue manager processes on all instances of a queue manager under the same domain local mqm group. The task, “Create a multi-instance queue manager on domain controllers” on page 1087, demonstrates how to set up a multi-instance queue manager using files secured in this way.

Configuration information

Configure as many queue manager instances as you need by modifying the IBM WebSphere MQ queue manager configuration information about each server. Each server must have the same version of IBM WebSphere MQ installed at a compatible fix level. The commands, **dspmqinf** and **addmqinf** assist you to configure the additional queue manager instances. Alternatively, you can edit the mqs.ini and qm.ini files directly. The topics, “Create a multi-instance queue manager on Linux” on page 1108, “Create a multi-instance queue manager on domain workstations or servers” on page 1073, and “Create a multi-instance queue manager on domain controllers” on page 1087 are examples showing how to configure a multi-instance queue manager.

On Windows, UNIX and Linux systems, you can share a single `mqs.ini` file by placing it on the network share and setting the **`AMQ_MQS_INI_LOCATION`** environment variable to point to it.

Restrictions

1. Configure multiple instances of the same queue manager only on servers having the same operating system, architecture and endianness. For example, both machines must be either 32-bit or 64-bit.
2. All IBM WebSphere MQ installations must be at release level 7.0.1 or higher.
3. Typically, active and standby installations are maintained at the same maintenance level. Consult the maintenance instructions for each upgrade to check whether you must upgrade all installations together.

   Note that the maintenance levels for the active and passive queue managers must be identical.
4. Share queue manager data and logs only between queue managers that are configured with the same IBM WebSphere MQ user, group, and access control mechanism.
5. On UNIX and Linux systems, configure the shared file system on networked storage with a `hard`, interruptible, mount rather than a `soft` mount. A hard interruptible mount forces the queue manager to hang until it is interrupted by a system call. Soft mounts do not guarantee data consistency after a server failure.
6. The shared log and data directories cannot be stored on a FAT, or an NFSv3 file system. For multi-instance queue managers on Windows, the networked storage must be accessed by the Common Internet File System (CIFS) protocol used by Windows networks.

Windows domains and multi-instance queue managers:

A multi-instance queue manager on Windows requires its data and logs to be shared. The share must be accessible to all instances of the queue manager running on different servers or workstations. Configure the queue managers and share as part of a Windows domain. The queue manager can run on a domain workstation or server, or on the domain controller.

Before configuring a multi-instance queue manager, read “Secure unshared queue manager data and log directories and files on Windows” on page 1099 and “Secure shared queue manager data and log directories and files on Windows” on page 1096 to review how to control access to queue manager data and log files. The topics are educational; if you want to go directly to setting up shared directories for a multi-instance queue manager in a Windows domain; see “Create a multi-instance queue manager on domain workstations or servers” on page 1073.

Run a multi-instance queue manager on domain workstations or servers

From Version 7.1, multi-instance queue managers run on a workstation or server that is a member of a domain. Before Version 7.1, multi-instance queue managers ran only on domain controllers; see “Run a multi-instance queue manager on domain controllers” on page 1073. To run a multi-instance queue manager on Windows, you require a domain controller, a file server, and two workstations or servers running the same queue manager connected to the same domain.

The change that makes it possible to run a multi-instance queue manager on any server or workstation in a domain, is that you can now create a queue manager with an additional security group. The additional security group is passed in the **`crtmqm`** command, in the **`-a`** parameter. You secure the directories that contain the queue manager data and logs with the group. The user ID that runs queue manager processes must be a member of this group. When the queue manager accesses the directories, Windows checks the permissions the user ID has to access the directories. By giving both the group and the user ID domain scope, the user ID running the queue manager processes has credentials from the global group. When the queue manager is running on a different server, the user ID running the queue manager processes can have the same credentials. The user ID does not have to be the same. It has to be a member of the alternative security group, as well as a member of the local `mqm` group.

The task of creating a multi-instance queue manager is the same as in Version 7.0.1 with one change. You must add the additional security group name to the parameters of the **crtmqm** command. The task is described in "Create a multi-instance queue manager on domain workstations or servers."

Multiple steps are required to configure the domain, and the domain servers and workstations. You must understand how Windows authorizes access by a queue manager to its data and log directories. If you are not sure how queue manager processes are authorized to access their log and data files read the topic "Secure unshared queue manager data and log directories and files on Windows" on page 1099. The topic includes two tasks to help you understand the steps the required. The tasks are "Reading and writing data and log files authorized by the local mqm group" on page 1101 and "Reading and writing data and log files authorized by an alternative local security group" on page 1104. Another topic, "Secure shared queue manager data and log directories and files on Windows" on page 1096, explains how to secure shared directories containing queue manager data and log files with the alternative security group. The topic includes four tasks, to set up a Windows domain, create a file share, install IBM WebSphere MQ for Windows, and configure a queue manager to use the share. The tasks are as follows:

1. "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076.
2. "Installing IBM WebSphere MQ on a server or workstation in a Windows domain" on page 1079.
3. "Creating a shared directory for queue manager data and log files" on page 1082.
4. "Reading and writing shared data and log files authorized by an alternative global security group" on page 1084.

You can then do the task, "Create a multi-instance queue manager on domain workstations or servers," using the domain. Do these tasks to explore setting up a multi-instance queue manager before transferring your knowledge to a production domain.

Run a multi-instance queue manager on domain controllers

In Version 7.0.1, multi-instance queue managers ran only on domain controllers. Queue manager data could be secured with the domain mqm group. As the topic "Secure shared queue manager data and log directories and files on Windows" on page 1096 explains, you cannot share directories secured with the local mqm group on workstations or servers. However on domain controllers all group and principals have domain scope. If you install IBM WebSphere MQ for Windows on a domain controller, the queue manager data and log files are secured with the domain mqm group, which can be shared. Follow the steps in the task, "Create a multi-instance queue manager on domain controllers" on page 1087 to configure a multi-instance queue manager on domain controllers.

Related information:

Managing Authorization and Access Control

Windows 2000, Windows Server 2003, and Windows Server 2008 cluster nodes as domain controllers

Create a multi-instance queue manager on domain workstations or servers:

An example shows how to set up a multi-instance queue manager on Windows on a workstation or a server that is part of a Windows domain. The server does not have to be a domain controller. The setup demonstrates the concepts involved, rather than being production scale. The example is based on Windows Server 2008. The steps might differ on other versions of Windows Server.

In a production scale configuration, you might have to tailor the configuration to an existing domain. For example, you might define different domain groups to authorize different shares, and to group the user IDs that run queue managers.

The example configuration consists of three servers:

sun A Windows Server 2008 domain controller. It owns the *wmq.example.com* domain that contains *Sun*, *mars*, and *venus*. For the purposes of illustration, it is also used as the file server.

mars A Windows Server 2008 used as the first IBM WebSphere MQ server. It contains one instance of the multi-instance queue manager called *QMGR*.

venus A Windows Server 2008 used as the second IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

Replace the italicized names in the example, with names of your choosing.

Before you begin

On Windows, you do not need to verify the file system that you plan to store queue manager data and log files on. The checking procedure, Verifying shared file system behavior, is applicable to UNIX and Linux. On Windows, the checks are always successful.

Do the steps in the following tasks. The tasks create the domain controller and domain, install IBM WebSphere MQ for Windows on one server, and create the file share for data and log files. If you are configuring an existing domain controller, you might find it useful to try out the steps on a new Windows Server 2008. You can adapt the steps to your domain.

1. "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076.
2. "Installing IBM WebSphere MQ on a server or workstation in a Windows domain" on page 1079.
3. "Creating a shared directory for queue manager data and log files" on page 1082.
4. "Reading and writing shared data and log files authorized by an alternative global security group" on page 1084.

About this task

This task is one of a sequence of tasks to configure a domain controller and two servers in the domain to run instances of a queue manager. In this task you configure a second server, *venus*, to run another instance of the queue manager *QMGR*. Follow the steps in this task to create the second instance of the queue manager, *QMGR*, and test that it works.

This task is separate from the four tasks in the preceding section. It contains the steps that convert a single instance queue manager into a multi-instance queue manager. All the other steps are common to single or multi-instance queue managers.

Procedure

1. Configure a second server to run IBM WebSphere MQ for Windows.
   a. Do the steps in the task "Installing IBM WebSphere MQ on a server or workstation in a Windows domain" on page 1079 to create a second domain server. In this sequence of tasks the second server is called *venus*.

      Tip: Create the second installation using the same installation defaults for IBM WebSphere MQ on each of the two servers. If the defaults differ, you might have to tailor the `Prefix` and the `InstallationName` variables in the *QMGR* **QueueManager** stanza in the IBM WebSphere MQ configuration file `mqs.ini`. The variables refer to paths that can differ for each installation and queue manager on each server. If the paths remain the same on every server, it is simpler to configure a multi-instance queue manager.
2. Create a second instance of *QMGR* on *venus*.
   a. If *QMGR* on *mars* does not exist, do the task "Reading and writing shared data and log files authorized by an alternative global security group" on page 1084, to create it
   b. Check the values of the `Prefix` and `InstallationName` parameters are correct for *venus*.

      On *mars*, run the **dspmqinf** command:

      ```
      dspmqinf QMGR
      ```

The system response:

```
QueueManager:
   Name=QMGR
   Directory=QMGR
   Prefix=C:\Program Files\IBM\WebSphere MQ
   DataPath=\\sun\wmq\data\QMGR
   InstallationName=Installation1
```

c. Copy the machine-readable form of the **QueueManager** stanza to the clipboard.

   On *mars* run the **dspmqinf** command again, with the **-o command** parameter.

   ```
   dspmqinf -o command QMGR
   ```

   The system response:

   ```
   addmqinf -s QueueManager -v Name=QMGR
   -v Directory=QMGR -v Prefix="C:\Program Files\IBM\WebSphere MQ"
   -v DataPath=\\sun\wmq\data\QMGR
   ```

d. On *venus* run the **addmqinf** command from the clipboard to create an instance of the queue manager on *venus*.

   Adjust the command, if necessary, to accommodate differences in the `Prefix` or `InstallationName` parameters.

   ```
   addmqinf -s QueueManager -v Name=QMGR
   -v Directory=QMGR -v Prefix="C:\Program Files\IBM\WebSphere MQ"
   -v DataPath=\\sun\wmq\data\QMGR

   WebSphere MQ configuration information added.
   ```

3. Start the queue manager *QMGR* on *venus*, permitting standby instances.

   a. Check *QMGR* on *mars* is stopped.

      On *mars*, run the **dspmq** command:

      ```
      dspmq -m QMGR
      ```

      The system response depends on how the queue manager was stopped; for example:

      ```
      C:\Users\Administrator>dspmq -m QMGR
      QMNAME(QMGR) STATUS(Ended immediately)
      ```

   b. On *venus* run the **strmqm** command to start *QMGR* permitting standbys:

      ```
      strmqm -x QMGR
      ```

      The system response:

      ```
      WebSphere MQ queue manager 'QMGR' starting.
      The queue manager is associated with installation 'Installation1'.
      5 log records accessed on queue manager 'QMGR' during the log
      replay phase.
      Log replay for queue manager 'QMGR' complete.
      Transaction manager state recovered for queue manager 'QMGR'.
      WebSphere MQ queue manager 'QMGR' started using V7.1.0.0.
      ```

Results

To test the multi-instance queue manager switches over, do the following steps:

1. On *mars*, run the **strmqm** command to start *QMGR* permitting standbys:

   ```
   strmqm -x QMGR
   ```

The system response:

```
WebSphere MQ queue manager 'QMGR' starting.
The queue manager is associated with installation 'Installation1'.
A standby instance of queue manager 'QMGR' has been started.
The active instance is running elsewhere.
```

2. On *venus* run the **endmqm** command:

```
endmqm -r -s -i QMGR
```

The system response on *venus*:

```
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ended, permitting switchover to
a standby instance.
```

And on *mars*:

```
dspmq
QMNAME(QMGR) STATUS(Running as standby)
C:\Users\wmquser2>dspmq
QMNAME(QMGR) STATUS(Running as standby)
C:\Users\wmquser2>dspmq
QMNAME(QMGR)  STATUS(Running)
```

What to do next

To verify a multi-instance queue manager using sample programs; see "Verify the multi-instance queue manager on Windows" on page 1094.

Creating an Active Directory and DNS domain for IBM WebSphere MQ:

This task creates the domain *wmq.example.com* on a Windows 2008 domain controller called *sun*. It configures the `Domain mqm` global group in the domain, with the correct rights, and with one user.

In a production scale configuration, you might have to tailor the configuration to an existing domain. For example, you might define different domain groups to authorize different shares, and to group the user IDs that run queue managers.

The example configuration consists of three servers:

sun A Windows Server 2008 domain controller. It owns the *wmq.example.com* domain that contains *Sun*, *mars*, and *venus*. For the purposes of illustration, it is also used as the file server.

mars A Windows Server 2008 used as the first IBM WebSphere MQ server. It contains one instance of the multi-instance queue manager called *QMGR*.

venus A Windows Server 2008 used as the second IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

Replace the italicized names in the example, with names of your choosing.

Before you begin

1. The task steps are consistent with a Windows Server 2008 that is installed but not configured with any roles. If you are configuring an existing domain controller, you might find it useful to try out the steps on a new Windows Server 2008. You can adapt the steps to your domain.

About this task

In this task, you create an Active Directory and DNS domain on a new domain controller. You then configure it ready to install IBM WebSphere MQ on other servers and workstations that join the domain. Follow the task if you are unfamiliar with installing and configuring Active Directory to create a Windows domain. You must create a Windows domain in order to create a multi-instance queue manager configuration. The task is not intended to guide you in the best way to configure a Windows domain. To deploy multi-instance queue managers in a production environment, you must consult Windows documentation.

During the task you do the following steps:
1. Install Active Directory.
2. Add a domain.
3. Add the domain to DNS.
4. Create the global group `Domain mqm` and give it the correct rights.
5. Add a user and make it a member of the global group `Domain mqm`.

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, “Windows domains and multi-instance queue managers” on page 1072.

For the purposes of the task the domain controller hostname is *`sun`*, and the two IBM WebSphere MQ servers are called *`mars`* and *`venus`*. The domain is called *`wmq.example.com`*. You can replace all the italicized names in the task with names of your own choosing.

Procedure

1. Log on to the domain controller, *`sun`*, as the local or `Workgroup` administrator.

   If the server is already configured as a domain controller, you must log on as a domain administrator.
2. Run the Active Directory Domain Services wizard.
   a. Click **Start** > **Run...** Type `dcpromo` and click **OK**.

   If the Active Directory binary files are not already installed, Windows installs the files automatically.
3. In the first window of the wizard, leave the **Use advanced mode installation** check box clear. Click **Next** > **Next** and click **Create a new domain in a new forest** > **Next**.
4. Type *`wmq.example.com`* into the **FQDN of the forest root domain** field. Click **Next**.
5. In the Set Forest Functional Level window, select **Windows Server 2003**, or later, from the list of **Forest functional levels** > **Next**.

   The oldest level of Windows Server that is supported by IBM WebSphere MQ is Windows Server 2003.
6. Optional: In the Set Domain Functional Level window, select **Windows Server 2003**, or later, from the list of **Domain functional levels** > **Next**.

   This step is only required if you set the Forest Functional Level to **Windows Server 2003**.
7. The Additional Domain Controller Options window opens, with **DNS server** selected as an additional option. Click **Next** and **Yes** to clear the warning window.

Tip: If a DNS server is already installed this option is not presented to you. If you want to follow this task precisely, remove all the roles from this domain controller and start again.

8. Leave the `Database`, `Log Files`, and `SYSVOL` directories unchanged; click **Next**.
9. Type a password into the **Password** and **Confirm password** fields in the Directory Services Restore Mode Administrator Password window. Click **Next** > **Next**. Select **Reboot on completion** in the final wizard window.
10. When the domain controller reboots, log on as *wmq*\Adminstrator.

    The server manager starts automatically.
11. Open the *wmq.example.com*\Users folder
    a. Open **Server Manager** > **Roles** > **Active Directory Domain Services** > ***wmq.example.com*** > **Users**.
12. Right-click **Users** > **New** > **Group**.
    a. Type a group name into the **Group name** field.

    Note: The preferred group name is `Domain mqm`. Type it exactly as shown.

    - Calling the group `Domain mqm` modifies the behavior of the “Prepare IBM WebSphere MQ” wizard on a domain workstation or server. It causes the “Prepare IBM WebSphere MQ” wizard automatically to add the group `Domain mqm` to the local `mqm` group on each new installation of IBM WebSphere MQ in the domain.
    - You can install workstations or servers in a domain with no `Domain mqm` global group. If you do so, you must define a group with the same properties as `Domain mqm` group. You must make that group, or the users that are members of it, members of the local `mqm` group wherever IBM WebSphere MQ is installed in a domain. You can place domain users into multiple groups. Create multiple domain groups, each group corresponding to a set of installations that you want to manage separately. Split domain users, according to the installations they manage, into different domain groups. Add each domain group or groups to the local `mqm` group of different IBM WebSphere MQ installations. Only domain users in the domain groups that are members of a specific local `mqm` group can create, administer, and run queue managers for that installation.
    - The domain user that you nominate when installing IBM WebSphere MQ on a workstation or server in a domain must be a member of the `Domain mqm` group, or of an alternative group you defined with same properties as the `Domain mqm` group.

    b. Leave **Global** clicked as the **Group scope**, or change it to **Universal**. Leave **Security** clicked as the **Group type**. Click **OK**.
13. Add the rights, **Allow Read group membership** and **Allow Read groupMembershipSAM** to the rights of the `Domain mqm` global group.
    a. In the Server Manager action bar, click **View** > **Advanced features**
    b. In the Server Manager navigation tree, click **Users**
    c. In the Users window, right-click **Domain mqm** > **Properties**
    d. Click **Security** > **Advanced** > **Add...**. Type `Domain mqm` and click **Check names** > **OK**.

    The **Name** field is prefilled with the string, `Domain mqm (`*domain name*`\Domain mqm)`.

    e. Click **Properties**. In the **Apply to** list, select **Descendant User Objects** from the bottom of the list.
    f. From the **Permissions** list, select the **Read group membership** and **Read groupMembershipSAM Allow** check boxes; click **OK** > **Apply** > **OK** > **OK**.
14. Add two or more users to the `Domain mqm` global group.

    One user, *wmquser1* in the example, runs the IBM IBM WebSphere MQ service, and the other user, *wmquser2*, is used interactively.

    A domain user is required to create a queue manager that uses the alternative security group in a domain configuration. It is not sufficient for the user ID to be an administrator, although an administrator has authority to run the **crtmqm** command. The domain user, who could be an administrator, must be a member of the local mqm group as well as of the alternative security group.

In the example, you make *wmquser1* and *wmquser2* members of the Domain mqm global group. The "Prepare IBM WebSphere MQ" wizard automatically configures Domain mqm as a member of the local mqm group where ever the wizard is run.

You must provide a different user to run the IBM IBM WebSphere MQ service for each installation of IBM WebSphere MQ on a single computer. You can reuse the same users on different computers.

a. In the Server Manager navigation tree, click **Users** > **New** > **User**

b. In the New Object - User window, type *wmquser1* into the **User logon name** field. Type *WebSphere* into the **First name** field, and *MQ1* into the **Last name** field. Click **Next**.

c. Type a password into the **Password** and **Confirm password** fields, and clear the **User must change password at next logon** check box. Click **Next** > **Finish**.

d. In the Users window, right-click ***WebSphere MQ*** > **Add to a group...**. Type Domain mqm and click **Check Names** > **OK** > **OK**.

e. Repeat steps a to d to add *WebSphere MQ2* as *wmquser2*.

15. Running IBM WebSphere MQ as a service. If you need to run IBM WebSphere MQ as a service, and then give the domain user (that you obtained from your domain administrator) the right to run as a service, carry out the following procedure:

a. Click **Start > Run...**. Type the command secpol.msc and click **OK**.

b. Open **Security Settings > Local Policies > User Rights Assignments**. In the list of policies, right-click **Log on as a service > Properties**.

c. Click **Add User or Group...** Type the name of the user you obtained from your domain administrator, and click **Check Names**

d. If prompted by a Windows Security window, type the user name and password of an account user or administrator with sufficient authority, and click **OK > Apply > OK**. Close the Local Security Policy window.

Note: On Windows Vista and Windows Server 2008 the User Account Control (UAC) is enabled by default.

The UAC feature restricts the actions users can perform on certain operating system facilities, even if they are members of the Administrators group. You must take appropriate steps to overcome this restriction.

What to do next

Proceed to the next task, "Installing IBM WebSphere MQ on a server or workstation in a Windows domain."

Installing IBM WebSphere MQ on a server or workstation in a Windows domain:

In this task, you install and configure IBM WebSphere MQ on a server or workstation in the *wmq.example.com* Windows domain.

In a production scale configuration, you might have to tailor the configuration to an existing domain. For example, you might define different domain groups to authorize different shares, and to group the user IDs that run queue managers.

The example configuration consists of three servers:

sun A Windows Server 2008 domain controller. It owns the *wmq.example.com* domain that contains *Sun*, *mars*, and *venus*. For the purposes of illustration, it is also used as the file server.

mars A Windows Server 2008 used as the first IBM WebSphere MQ server. It contains one instance of the multi-instance queue manager called *QMGR*.

venus A Windows Server 2008 used as the second IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

Replace the italicized names in the example, with names of your choosing.

Before you begin

1. Do the steps in "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076 to create a domain controller, *sun*, for the domain *wmq.example.com*. Change the italicized names to suit your configuration.
2. See Hardware and software requirements on Windows systems for other Windows versions you can run IBM WebSphere MQ on.

About this task

In this task you configure a Windows Server 2008, called *mars*, as a member of the *wmq.example.com* domain. You install IBM WebSphere MQ, and configure the installation to run as a member of the *wmq.example.com* domain.

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, "Windows domains and multi-instance queue managers" on page 1072.

For the purposes of the task the domain controller hostname is *sun*, and the two IBM WebSphere MQ servers are called *mars* and *venus*. The domain is called *wmq.example.com*. You can replace all the italicized names in the task with names of your own choosing.

Procedure

1. Add the domain controller, *sun.wmq.example.com* to *mars* as a DNS server.
   a. On *mars*, log on as *mars*\Administrator and click **Start**.
   b. Right-click **Network** > **Properties** > **Manage network connections**.
   c. Right-click the network adapter, click **Properties**.

      The system responds with the Local Area Connection Properties window listing items the connection uses.
   d. Select the **Internet Protocol Version 4** or **Internet Protocol Version 6** from the list of items in the Local Area Connection Properties window. Click **Properties** > **Advanced...** and click the **DNS** tab.
   e. Under the DNS server addresses, click **Add...**.
   f. Type the IP address of the domain controller, which is also the DNS server, and click **Add**.
   g. Click **Append these DNS suffixes** > **Add...**.
   h. Type *wmq.example.com* and click **Add**.
   i. Type *wmq.example.com* in the **DNS suffix for this connection** field.
   j. Select **Register this connection's address in DNS** and **Use this connection's suffix in DNS registration**. Click **OK** > **OK** > **Close**
   k. Open a command window, and type the command **ipconfig /all** to review the TCP/IP settings.
2. On *mars*, add the computer to the *wmq.example.com* domain.
   a. Click **Start**
   b. Right-click **Computer** > **Properties**. In the Computer name, domain and workgroup settings division, click **Change settings**.
   c. In the System Properties windows, click **Change...**.

d. Click Domain, type *wmq.example.com*, and click **OK**.

e. Type the **User name** and **Password** of the domain controller administrator, who has the authority to permit the computer to join the domain, and click **OK**.

f. Click **OK** > **OK** > **Close** > **Restart Now** in response to the "Welcome to the *wmq.example.com* domain" message.

3. Check that the computer is a member of the *wmq.example.com* domain

   a. On *sun*, log on to the domain controller as *wmq*\Administrator.

   b. Open **Server Manager** > **Active Directory Domain Services** > ***wmq.example.com*** > **Computers** and check *mars* is listed correctly in the Computers window.

4. Install IBM WebSphere MQ for Windows on *mars*.

   For further information about running the IBM WebSphere MQ for Windows installation wizard; see Installing IBM WebSphere MQ server on Windows.

   a. On *mars*, log on as the local administrator, *mars*\Administrator.

   b. Run the **Setup** command on the IBM WebSphere MQ for Windows installation media.

      The IBM WebSphere MQ Launchpad application starts.

   c. Click **Software Requirements** to check that the prerequisite software is installed.

   d. Click **Network Configuration** > **Yes** to configure a domain user ID.

      The task, "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076, configures a domain user ID for this set of tasks.

   e. Click **WebSphere MQ Installation**, select an installation language and click Launch IBM IBM WebSphere MQ Installer.

   f. Confirm the license agreement and click **Next** > **Next** > **Install** to accept the default configuration. Wait for the installation to complete, and click **Finish**.

      You might want to change the name of the installation, install different components, configure a different directory for queue manager data and logs, or install into a different directory. If so, click **Custom** rather than **Typical**. IBM WebSphere MQ is installed, and the installer starts the "Prepare IBM WebSphere MQ" wizard.

      Important: Do not run the wizard yet.

5. Configure the user that is going to run the IBM IBM WebSphere MQ service with the **Run as a service** right.

   Choose whether to configure the local mqm group, the Domain mqm group, or the user that is going to run the IBM IBM WebSphere MQ service with the right. In the example, you give the user the right.

   a. Click **Start** > **Run...**, type the command **secpol.msc** and click **OK**.

   b. Open **Security Settings** > **Local Policies** > **User Rights Assignments**. In the list of policies, right-click **Log on as a service** > **Properties**.

   c. Click **Add User or Group...** and type *wmquser1* and click **Check Names**

   d. Type the user name and password of a domain administrator, *wmq*\Administrator, and click **OK** > **Apply** > **OK**. Close the Local Security Policy window.

6. Run the "Prepare IBM WebSphere MQ" wizard.

   For further information about running the "Prepare IBM WebSphere MQ" wizard; see Configuring WebSphere MQ with the Prepare WebSphere MQ wizard.

   a. The IBM IBM WebSphere MQ Installer runs the "Prepare IBM WebSphere MQ" automatically.

      To start the wizard manually, find the shortcut to the "Prepare IBM WebSphere MQ" in the **Start** > **All programs** > **IBM WebSphere MQ** folder. Select the shortcut that corresponds to the installation of IBM WebSphere MQ in a multi-installation configuration.

   b. Click **Next** and leave **Yes** clicked in response to the question "Identify if there is a Windows 2000 or later domain controller in the network".

c. Click **Yes** > **Next** in the first Configuring IBM WebSphere MQ for Windows for Windows domain users window.

d. In the second Configuring IBM WebSphere MQ for Windows for Windows domain users window, type `wmq` in the **Domain** field. Type `wmquser1` in the **User name** field, and the password, if you set one, in the **Password** field. Click **Next**.

   The wizard configures and starts the IBM IBM WebSphere MQ with *wmquser1*.

e. In the final page of the wizard, select or clear the check boxes as you require and click **Finish**.

What to do next

1. Do the task, "Reading and writing data and log files authorized by the local `mqm` group" on page 1101, to verify that the installation and configuration are working correctly.
2. Do the task, "Creating a shared directory for queue manager data and log files," to configure a file share to store the data and log files of a multi-instance queue manager.

Related information:

User rights required for a WebSphere MQ Windows Service

Creating a shared directory for queue manager data and log files:

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing.

In a production scale configuration, you might have to tailor the configuration to an existing domain. For example, you might define different domain groups to authorize different shares, and to group the user IDs that run queue managers.

The example configuration consists of three servers:

`sun` A Windows Server 2008 domain controller. It owns the *`wmq.example.com`* domain that contains *`Sun`*, *`mars`*, and *`venus`*. For the purposes of illustration, it is also used as the file server.

`mars` A Windows Server 2008 used as the first IBM WebSphere MQ server. It contains one instance of the multi-instance queue manager called *`QMGR`*.

`venus` A Windows Server 2008 used as the second IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *`QMGR`*.

Replace the italicized names in the example, with names of your choosing.

Before you begin

1. To do this task exactly as documented, do the steps in the task, "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076, to create the domain *`sun.wmq.example.com`* on the domain controller *`sun`*. Change the italicized names to suit your configuration.

About this task

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, "Windows domains and multi-instance queue managers" on page 1072.

In the task, you create a share containing a data and log directory, and a global group to authorize access to the share. You pass the name of the global group that authorizes the share to the **`crtmqm`** command in its **-a** parameter. The global group gives you the flexibility of separating the users of this share from users of other shares. If you do not need this flexibility, authorize the share with the `Domain mqm` group rather than create a new global group.

The global group used for sharing in this task is called *wmqha*, and the share is called *wmq*. They are defined on the domain controller *sun* in the Windows domain *wmq.example.com*. The share has full control permissions for the global group *wmqha*. Replace the italicized names in the task with names of your choosing.

For the purposes of this task the domain controller is the same server as the file server. In practical applications, split the directory and file services between different servers for performance and availability.

You must configure the user ID that the queue manager is running under to be a member of two groups. It must be a member of the local mqm group on a IBM WebSphere MQ server, and of the *wmqha* global group.

In this set of tasks, when the queue manager is running as a service, it runs under the user ID *wmquser1*, so *wmquser1* must be a member of *wmqha*. When the queue manager is running interactively, it runs under the user ID *wmquser2*, so *wmquser2* must be a member of *wmqha*. Both *wmquser1* and *wmquser2* are members of the global group Domain mqm. Domain mqm is a member of the local mqm group on the *mars* and *venus* IBM WebSphere MQ servers. Hence, *wmquser1* and *wmquser2* are members of the local mqm group on bothIBM WebSphere MQ servers.

Procedure

1. Log on to the domain controller, *sun.wmq.example.com* as the domain administrator.
2. Create the global group *wmqha*.
   a. Open **Server Manager** > **Roles** > **Active Directory Domain Services** > ***wmq.example.com*** > **Users**.
   b. Open the *wmq.example.com*\Users folder
   c. Right-click **Users** > **New** > **Group**.
   d. Type *wmqha* into the **Group name** field.
   e. Leave **Global** clicked as the **Group scope** and **Security** as the **Group type**. Click **OK**.
3. Add the domain users *wmquser1* and *wmquser2* to the global group, *wmqha*.
   a. In the Server Manager navigation tree, click **Users** and right-click ***wmqha*** > **Properties** in the list of users.
   b. Click the Members tab in the *wmqha* Properties window.
   c. Click **Add...**; type *wmquser1*; *wmquser2* and click **Check Names** > **OK** > **Apply** > **OK**.
4. Create the directory tree to contain queue manager data and log files.
   a. Open a command prompt.
   b. Type the command:

   ```
   md c:\wmq\data , c:\wmq\logs
   ```
5. Authorize the global group *wmqha* to have full control permission to the *c:\wmq* directories and share.
   a. In Windows Explorer, right-click ***c:\wmq*** > **Properties**.
   b. Click the **Security** tab and click **Advanced** > **Edit...**.
   c. Clear the check box for **Include inheritable permissions from this object's owner**. Click **Copy** in the Windows Security window.
   d. Select the lines for Users in the list of **Permission entries** and click **Remove**. Leave the lines for SYSTEM, Administrators, and CREATOR OWNER in the list of **Permission entries**.
   e. Click **Add...**, and type the name of the global group *wmqha*. Click **Check Names** > **OK**.
   f. In the Permission Entry for wmq window, select **Full Control** in the list of **Permissions**.
   g. Click **OK** > **Apply** > **OK** > **OK** > **OK**
   h. In Windows Explorer, right-click ***c:\wmq*** > **Share...**.
   i. Click **Advanced Sharing...** and select the **Share this folder** check box. Leave the share name as *wmq*.

j. Click **Permissions** > **Add...**, and type the name of the global group *wmqha*. Click **Check Names** > **OK**.

k. Select *wmqha* in the list of **Group or user names**. Select the **Full Control** check box in the list of **Permissions for *wmqha***; click **Apply**.

l. Select *Administrators* in the list of **Group or user names**. Select the **Full Control** check box in the list of **Permissions for *Administrators***; click **Apply** > **OK** > **OK** > **Close**.

What to do next

Check that you can read and write files to the shared directories from each of the IBM WebSphere MQ servers. Check the IBM IBM WebSphere MQ service user ID, *wmquser1* and the interactive user ID, *wmquser2*.

1. If you are using remote desktop, you must add *wmq\wmquser1* and *wmquser2* to the local group Remote Desktop Users on *mars*.
   a. Log on to *mars* as *wmq*\Administrator
   b. Run the **lusrmgr.msc** command to open the Local Users and Groups window.
   c. Click **Groups**. Right-click **Remote Desktop Users** > **Properties** > **Add...**. Type *wmquser1* **;** *wmquser2* and click **Check Names**.
   d. Type in the user name and password of the domain administrator, *wmq*\Administrator, and click **OK** > **Apply** > **OK**.
   e. Close the Local Users and Groups window.
2. Log on to *mars* as *wmq\wmquser1*.
   a. Open a Windows Explorer window, and type in \\*sun**wmq*.

      The system responds by opening the *wmq* share on *sun.wmq.example.com*, and lists the data and logs directories.
   b. Check the permissions of *wmquser1* by creating a file in data subdirectory, adding some content, reading it, and then deleting it.
3. Log on to *mars* as *wmq\wmquser2*, and repeat the checks.
4. Do the next task, to create a queue manager to use the shared data and log directories; see "Reading and writing shared data and log files authorized by an alternative global security group."

Reading and writing shared data and log files authorized by an alternative global security group:

This task shows how to use the **-a** flag on the **crtmqm** command. The **-a** flag gives the queue manager access to its log and data files on a remote file share using the alternative security group.

In a production scale configuration, you might have to tailor the configuration to an existing domain. For example, you might define different domain groups to authorize different shares, and to group the user IDs that run queue managers.

The example configuration consists of three servers:

sun A Windows Server 2008 domain controller. It owns the *wmq.example.com* domain that contains *Sun*, *mars*, and *venus*. For the purposes of illustration, it is also used as the file server.

mars A Windows Server 2008 used as the first IBM WebSphere MQ server. It contains one instance of the multi-instance queue manager called *QMGR*.

venus A Windows Server 2008 used as the second IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

Replace the italicized names in the example, with names of your choosing.

Before you begin

Do the steps in the following tasks. The tasks create the domain controller and domain, install IBM WebSphere MQ for Windows on one server, and create the file share for data and log files. If you are configuring an existing domain controller, you might find it useful to try out the steps on a new Windows Server 2008. You can adapt the steps to your domain.

1. "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076.
2. "Installing IBM WebSphere MQ on a server or workstation in a Windows domain" on page 1079.
3. "Creating a shared directory for queue manager data and log files" on page 1082.

About this task

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, "Windows domains and multi-instance queue managers" on page 1072.

In this task, you create a queue manager that stores its data and logs in a remote directory on a file server. For the purposes of this example, the file server is the same server as the domain controller. The directory containing the data and log folders is shared with full control permission given to the global group `wmqha`.

Procedure

1. Log on to the domain server, *`mars`*, as the local administrator, *`mars`*`\Administrator`.
2. Open a command window.
3. Restart the IBM IBM WebSphere MQ service.

   You must restart the service so that the user ID it runs under acquires the additional security credentials you configured for it.

   Type the commands:

   ```
   endmqsvc
   strmqsvc
   ```

   The system responses:

   ```
   5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
   The MQ service for installation 'Installation1' ended successfully.
   ```

   And:

   ```
   5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
   The MQ service for installation 'Installation1' started successfully.
   ```
4. Create the queue manager.

   ```
   crtmqm -a wmq\wmqha -sax -u SYSTEM.DEAD.LETTER.QUEUE -md \\sun\wmq\data -ld \\sun\wmq\logs  QMGR
   ```

   You must specify the domain, *`wmq`*, of the alternative security group *`wmqha`* by specifying full domain name of the global group *`"wmq\wmqha"`*.

   You must spell out the Universal Naming Convention (UNC) name of the share `\\`*`sun\wmq`*, and not use a mapped drive reference.

   The system response:

   ```
   WebSphere MQ queue manager created.
   Directory '\\sun\wmq\data\QMGR' created.
   The queue manager is associated with installation '1'
   Creating or replacing default objects for queue manager 'QMGR'
   Default objects statistics : 74 created. 0 replaced.
   Completing setup.
   Setup completed.
   ```

What to do next

Test the queue manager by putting and getting a message to a queue.

1. Start the queue manager.

   ```
   strmqm QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' starting.
   The queue manager is associated with installation '1'.
   5 log records accessed on queue manager 'QMGR' during the log
   replay phase.
   Log replay for queue manager 'QMGR' complete.
   Transaction manager state recovered for queue manager 'QMGR'.
   WebSphere MQ queue manager 'QMGR' started using V7.1.0.0.
   ```

2. Create a test queue.

   ```
   echo define qlocal(QTEST) | runmqsc QMGR
   ```

   The system response:

   ```
   5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
   Starting MQSC for queue manager QMGR.


        1 : define qlocal(QTEST)
   AMQ8006: WebSphere MQ queue created.
   One MQSC command read.
   No commands have a syntax error.
   All valid MQSC commands were processed.
   ```

3. Put a test message using the sample program **amqsput**.

   ```
   echo 'A test message' | amqsput QTEST QMGR
   ```

   The system response:

   ```
   Sample AMQSPUT0 start
   target queue is QTEST
   Sample AMQSPUT0 end
   ```

4. Get the test message using the sample program **amqsget**.

   ```
   amqsget QTEST QMGR
   ```

   The system response:

   ```
   Sample AMQSGET0 start
   message <A test message>
   Wait 15 seconds ...
   no more messages
   Sample AMQSGET0 end
   ```

5. Stop the queue manager.

   ```
   endmqm -i QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' ending.
   WebSphere MQ queue manager 'QMGR' ended.
   ```

6. Delete the queue manager.

   ```
   dltmqm QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' deleted.
   ```
7. Delete the directories you created.

   Tip: Add the /Q option to the commands to prevent the command prompting to delete each file or directory.

   ```
   del /F /S C:\wmq\*.*
   rmdir /S C:\wmq
   ```

Create a multi-instance queue manager on domain controllers:

An example shows how to set up a multi-instance queue manager on Windows on domain controllers. The setup demonstrates the concepts involved, rather than being production scale. The example is based on Windows Server 2008. The steps might differ on other versions of Windows Server.

The configuration uses the concept of a mini-domain, or "domainlet"; see Windows 2000, Windows Server 2003, and Windows Server 2008 cluster nodes as domain controllers. To add multi-instance queue managers to an existing domain, see "Create a multi-instance queue manager on domain workstations or servers" on page 1073.

The example configuration consists of three servers:

sun A Windows Server 2008 server used as the first domain controller. It defines the *wmq.example.com* domain that contains *sun*, *earth*, and *mars*. It contains one instance of the multi-instance queue manager called *QMGR*.

earth A Windows Server 2008 used as the second domain controller IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

mars A Windows Server 2008 used as the file server.

Replace the italicized names in the example, with names of your choosing.

Before you begin

1. On Windows, you do not need to verify the file system that you plan to store queue manager data and log files on. The checking procedure, Verifying shared file system behavior, is applicable to UNIX and Linux. On Windows, the checks are always successful.
2. Do the steps in "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076 to create the first domain controller.
3. Do the steps in "Adding a second domain controller to the *wmq.example.com* domain" on page 1090 to add a second domain controller, install IBM WebSphere MQ for Windows on both domain controllers, and verify the installations.
4. Do the steps in "Installing WebSphere MQ on domain controllers in the *wmq.example.com* domain" on page 1092 to install IBM WebSphere MQ on the two domain controllers.

About this task

On a file server in the same domain create a share for the queue manager log and data directories. Next, create the first instance of a multi-instance queue manager that uses the file share on one of the domain controllers. Create the other instance on the other domain controller and finally verify the configuration. You can create the file share on a domain controller.

In the sample, *sun* is the first domain controller, *earth* the second, and *mars* is the file server.

Procedure

1. Create the directories that are to contain the queue manager data and log files.
   a. On *mars*, type the command:
   ```
   md c:\wmq\data , c:\wmq\logs
   ```
2. Share the directories that are to contain the queue manager data and log files.

   You must permit full control access to the domain local group mqm, and the user ID you use to create the queue manager. In the example, user IDs that are members of Domain Administrators have the authority to create queue managers.

   The file share must be on a server that is in the same domain as the domain controllers. In the example, the server *mars* is in the same domain as the domain controllers.
   a. In Windows Explorer, right-click ***c:\wmq*** > **Properties**.
   b. Click the **Security** tab and click **Advanced** > **Edit...**.
   c. Clear the check box for **Include inheritable permissions from this object's owner**. Click **Copy** in the Windows Security window.
   d. Select the lines for Users in the list of **Permission entries** and click **Remove**. Leave the lines for SYSTEM, Administrators, and CREATOR OWNER in the list of **Permission entries**.
   e. Click **Add...**, and type the name of the domain local group *mqm*. Click **Check Names**
   f. In response to a Windows Security window, Type the name and password of the Domain Administrator and click **OK** > **OK**.
   g. In the Permission Entry for wmq window, select **Full Control** in the list of **Permissions**.
   h. Click **OK** > **Apply** > **OK** > **OK** > **OK**
   i. Repeat steps e to h to add Domain Administrators.
   j. In Windows Explorer, right-click ***c:\wmq*** > **Share...**.
   k. Click **Advanced Sharing...** and select the **Share this folder** check box. Leave the share name as *wmq*.
   l. Click **Permissions** > **Add...**, and type the name of the domain local group *mqm*; Domain Administrators. Click **Check Names**.
   m. In response to a Windows Security window, Type the name and password of the Domain Administrator and click **OK** > **OK**.
3. Create the queue manager *QMGR* on the first domain controller, *sun*.
   ```
   crtmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE -md \\mars\wmq\data -ld \\mars\wmq\logs  QMGR
   ```
   The system response:
   ```
   WebSphere MQ queue manager created.
   Directory '\\mars\wmq\data\QMGR' created.
   The queue manager is associated with installation 'Installation1'.
   Creating or replacing default objects for queue manager 'QMGR'.
   Default objects statistics : 74 created. 0 replaced. 0 failed.
   Completing setup.
   Setup completed.
   ```
4. Start the queue manager on *sun*, permitting a standby instance.
   ```
   strmqm -x QMGR
   ```
   The system response:
   ```
   WebSphere MQ queue manager 'QMGR' starting.
   The queue manager is associated with installation 'Installation1'.
   5 log records accessed on queue manager 'QMGR' during the log
   replay phase.
   ```

```
Log replay for queue manager 'QMGR' complete.
Transaction manager state recovered for queue manager 'QMGR'.
WebSphere MQ queue manager 'QMGR' started using V7.1.0.0.
```

5. Create a second instance of *QMGR* on *earth*.
   a. Check the values of the `Prefix` and `InstallationName` parameters are correct for *earth*.

      On *sun*, run the **dspmqinf** command:

      `dspmqinf QMGR`

      The system response:

      ```
      QueueManager:
         Name=QMGR
         Directory=QMGR
         Prefix=C:\Program Files\IBM\WebSphere MQ
         DataPath=\\mars\wmq\data\QMGR
         InstallationName=Installation1
      ```
   b. Copy the machine-readable form of the **QueueManager** stanza to the clipboard.

      On *sun* run the **dspmqinf** command again, with the **-o command** parameter.

      `dspmqinf -o command QMGR`

      The system response:

      ```
      addmqinf -s QueueManager -v Name=QMGR
      -v Directory=QMGR -v Prefix="C:\Program Files\IBM\WebSphere MQ"
      -v DataPath=\\mars\wmq\data\QMGR
      ```
   c. On *earth* run the **addmqinf** command from the clipboard to create an instance of the queue manager on *earth*.

      Adjust the command, if necessary, to accommodate differences in the `Prefix` or `InstallationName` parameters.

      ```
      addmqinf -s QueueManager -v Name=QMGR
      -v Directory=QMGR -v Prefix="C:\Program Files\IBM\WebSphere MQ"
      -v DataPath=\\mars\wmq\data\QMGR

      WebSphere MQ configuration information added.
      ```
6. Start the standby instance of the queue manager on *earth*.

   `strmqm -x QMGR`

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' starting.
   The queue manager is associated with installation 'Installation1'.
   A standby instance of queue manager 'QMGR' has been started. The active
   instance is running elsewhere.
   ```

Results

Verify that the queue manager switches over from *sun* to *earth*:

1. On *sun*, run the command:

   `endmqm -i -r -s QMGR`

   The system response on *sun*:

   ```
   WebSphere MQ queue manager 'QMGR' ending.
   WebSphere MQ queue manager 'QMGR' ending.
   WebSphere MQ queue manager 'QMGR' ending.
   ```

```
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ending.
WebSphere MQ queue manager 'QMGR' ended, permitting switchover to
a standby instance.
```

2. On *earth* repeatedly type the command:

   ```
   dspmq
   ```

   The system responses:

   ```
   QMNAME(QMGR) STATUS(Running as standby)
   QMNAME(QMGR) STATUS(Running as standby)
   QMNAME(QMGR) STATUS(Running)
   ```

What to do next

To verify a multi-instance queue manager using sample programs; see "Verify the multi-instance queue manager on Windows" on page 1094.

Related tasks:

"Adding a second domain controller to the *wmq.example.com* domain"
Add a second domain controller to the *wmq.example.com* domain to construct a Windows domain in which to run multi-instance queue managers on domain controllers and file servers.

"Installing WebSphere MQ on domain controllers in the *wmq.example.com* domain" on page 1092
Put your short description here; used for first paragraph and abstract.

Related information:

⇨ Windows 2000, Windows Server 2003, and Windows Server 2008 cluster nodes as domain controllers

Adding a second domain controller to the wmq.example.com domain:

Add a second domain controller to the *wmq.example.com* domain to construct a Windows domain in which to run multi-instance queue managers on domain controllers and file servers.

The example configuration consists of three servers:

sun A Windows Server 2008 server used as the first domain controller. It defines the *wmq.example.com* domain that contains *sun*, *earth*, and *mars*. It contains one instance of the multi-instance queue manager called *QMGR*.

earth A Windows Server 2008 used as the second domain controller IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

mars A Windows Server 2008 used as the file server.

Replace the italicized names in the example, with names of your choosing.

Before you begin

1. Do the steps in "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076 to create a domain controller, *sun*, for the domain *wmq.example.com*. Change the italicized names to suit your configuration.
2. Install Windows Server 2008 on a server in the default workgroup, WORKGROUP. For the example, the server is named *earth*.

About this task

In this task you configure a Windows Server 2008, called *earth*, as a second domain controller in the *wmq.example.com* domain.

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, “Windows domains and multi-instance queue managers” on page 1072.

Procedure

1. Add the domain controller, *sun.wmq.example.com* to *earth* as a DNS server.
   a. On *earth*, log on as *earth*\Administrator and click **Start**.
   b. Right-click **Network** > **Properties** > **Manage network connections**.
   c. Right-click the network adapter, click **Properties**.

      The system responds with the Local Area Connection Properties window listing items the connection uses.
   d. Select the **Internet Protocol Version 4** or **Internet Protocol Version 6** from the list of items in the Local Area Connection Properties window. Click **Properties** > **Advanced...** and click the **DNS** tab.
   e. Under the DNS server addresses, click **Add...**.
   f. Type the IP address of the domain controller, which is also the DNS server, and click **Add**.
   g. Click **Append these DNS suffixes** > **Add...**.
   h. Type *wmq.example.com* and click **Add**.
   i. Type *wmq.example.com* in the **DNS suffix for this connection** field.
   j. Select **Register this connection's address in DNS** and **Use this connection's suffix in DNS registration**. Click **OK** > **OK** > **Close**
   k. Open a command window, and type the command **ipconfig /all** to review the TCP/IP settings.
2. Log on to the domain controller, *sun*, as the local or Workgroup administrator.

   If the server is already configured as a domain controller, you must log on as a domain administrator.
3. Run the Active Directory Domain Services wizard.
   a. Click **Start** > **Run...** Type dcpromo and click **OK**.

   If the Active Directory binary files are not already installed, Windows installs the files automatically.
4. Configure *earth* as the second domain controller in the *wmq.example.com* domain.
   a. In the first window of the wizard, leave the **Use advanced mode installation** check box clear. Click **Next** > **Next** and click **Create Add a domain controller to an existing domain** > **Next**.
   b. Type *wmq* into the **Type the name of any domain in this forest ...** field. The **Alternate credentials** radio button is clicked, click **Set...**. Type in the name and password of the domain administrator and click **OK** > **Next** > **Next** > **Next**.
   c. In the Additional Domain Controller Options window accept the **DNS server** and **Global catalog** options, which are selected; click **Next** > **Next**.
   d. On the Directory Services Restore Mode Administrator Password, type in a **Password** and **Confirm password** and click **Next** > **Next**.
   e. When prompted for **Network Credentials**, type in the password of the domain administrator. Select **Reboot on completion** in the final wizard window.
   f. After a while, a window might open with a **DCPromo** error concerning DNS delegation; click **OK**. The server reboots.

Results

When *earth* has rebooted, log on as Domain Administrator. Check that the wmq.example.com domain has been replicated to *earth*.

What to do next

Continue with installing IBM WebSphere MQ; see "Installing WebSphere MQ on domain controllers in the *wmq.example.com* domain."

Related tasks:

"Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076
This task creates the domain *wmq.example.com* on a Windows 2008 domain controller called *sun*. It configures the Domain mqm global group in the domain, with the correct rights, and with one user.

Installing WebSphere MQ on domain controllers in the wmq.example.com domain:

Install and configure installations of IBM WebSphere MQ on both domain controllers in the *wmq.example.com* domain.

Put your short description here; used for first paragraph and abstract.

The example configuration consists of three servers:

sun A Windows Server 2008 server used as the first domain controller. It defines the *wmq.example.com* domain that contains *sun*, *earth*, and *mars*. It contains one instance of the multi-instance queue manager called *QMGR*.

earth A Windows Server 2008 used as the second domain controller IBM WebSphere MQ server. It contains the second instance of the multi-instance queue manager called *QMGR*.

mars A Windows Server 2008 used as the file server.

Replace the italicized names in the example, with names of your choosing.

Before you begin

1. Do the steps in "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076 to create a domain controller, *sun*, for the domain *wmq.example.com*. Change the italicized names to suit your configuration.
2. Do the steps in "Adding a second domain controller to the *wmq.example.com* domain" on page 1090 to create a second domain controller, *earth*, for the domain *wmq.example.com*. Change the italicized names to suit your configuration.
3. See Hardware and software requirements on Windows systems for other Windows versions you can run IBM WebSphere MQ on.

About this task

Install and configure installations of IBM WebSphere MQ on both domain controllers in the *wmq.example.com* domain.

Procedure

1. Install IBM WebSphere MQ on *sun* and *earth*.

   For further information about running the IBM WebSphere MQ for Windows installation wizard; see Installing IBM WebSphere MQ server on Windows .

   a. On both *sun* and *earth*, log on as the domain administrator, *wmq*\Administrator.
   b. Run the **Setup** command on the IBM WebSphere MQ for Windows installation media.

The IBM WebSphere MQ Launchpad application starts.

c. Click **Software Requirements** to check that the prerequisite software is installed.

d. Click **Network Configuration** > **No**.

You can configure either a domain user ID or not for this installation. The user ID that is created is a domain local user ID.

e. Click **WebSphere MQ Installation**, select an installation language and click Launch IBM IBM WebSphere MQ Installer.

f. Confirm the license agreement and click **Next** > **Next** > **Install** to accept the default configuration. Wait for the installation to complete, and click **Finish**.

If you want to change the name of the installation, install different components, configure a different directory for queue manager data and logs, or install into a different directory, click **Custom** rather than **Typical**. IBM WebSphere MQ is installed, and the installer starts the “Prepare IBM WebSphere MQ” wizard.

The IBM WebSphere MQ for Windows installation configures a domain local group mqm, and a domain group Domain mqm. It makes Domain mqm a member of mqm. Subsequent domain controllers in the same domain share the mqm and Domain mqm groups.

2. On both *earth* and *sun*, run the “Prepare IBM WebSphere MQ” wizard.

For further information about running the “Prepare IBM WebSphere MQ” wizard, see Configuring WebSphere MQ with the Prepare WebSphere MQ wizard .

a. The IBM WebSphere MQ installer runs the “Prepare IBM WebSphere MQ” automatically.

To start the wizard manually, find the shortcut to the “Prepare IBM WebSphere MQ” in the **Start** > **All programs** > **IBM WebSphere MQ** folder. Select the shortcut that corresponds to the installation of IBM WebSphere MQ in a multi-installation configuration.

b. Click **Next** and leave **No** clicked in response to the question “Identify if there is a Windows 2000 or later domain controller in the network”[11].

c. In the final page of the wizard, select or clear the check boxes as you require and click **Finish**.

The “Prepare IBM WebSphere MQ” wizard creates a domain local user MUSR_MQADMIN on the first domain controller, and another domain local user MUSR_MQADMIN1 on the second domain controller. The wizard creates the IBM IBM WebSphere MQ service on each controller, with MUSR_MQADMIN or MUSR_MQADMIN1 as the user that logs on the service.

3. Define a user that has permission to create a queue manager.

The user must have the right to log on locally, and be a member of the domain local mqm group. On domain controllers, domain users do not have the right to log on locally, but administrators do. By default, no user has both these attributes. In this task, add domain administrators to the domain local mqm group.

a. Open **Server Manager** > **Roles** > **Active Directory Domain Services** > ***wmq.example.com*** > **Users**.

b. Right-click **Domain Admins** > **Add to a group...** and type mqm; click **Check names** > **OK** > **OK**

Results

1. Check that the “Prepare IBM WebSphere MQ” created the domain user, MUSR_MQADMIN:

a. Open **Server Manager** > **Roles** > **Active Directory Domain Services** > ***wmq.example.com*** > **Users**.

b. Right-click **MUSR_MQADMIN** > **Properties...** > **Member Of**, and see that it is a member of Domain users and mqm.

2. Check that MUSR_MQADMIN has the right to run as a service:

a. Click Click **Start** > **Run...**, type the command **secpol.msc** and click **OK**.

[11] You can configure the installation for the domain. As all users and groups on a domain controller have domain scope, it does not make any difference. It is simpler to install IBM WebSphere MQ as if it is not in domain.

b. Open **Security Settings** > **Local Policies** > **User Rights Assignments**. In the list of policies, right-click **Log on as a service** > **Properties**, and see MUSR_MQADMIN is listed as having the right to log on as a service. Click **OK**.

What to do next

1. Do the task, "Reading and writing data and log files authorized by the local mqm group" on page 1101, to verify that the installation and configuration are working correctly.
2. Go back to the task, "Create a multi-instance queue manager on domain controllers" on page 1087, to complete the task of configuring a multi-instance queue manager on domain controllers.

Related information:

User rights required for a WebSphere MQ Windows Service

Verify the multi-instance queue manager on Windows:

Use the sample programs **amqsghac**, **amqsphac** and **amqsmhac** to verify a multi-instance queue manager configuration. This topic provides an example configuration to verify a multi-instance queue manager configuration on Windows Server 2003.

The high availability sample programs use automatic client reconnection. When the connected queue manager fails, the client attempts to reconnect to a queue manager in the same queue manager group. The description of the samples, High availability sample programs, demonstrates client reconnection using a single instance queue manager for simplicity. You can use the same samples with multi-instance queue managers to verify a multi-instance queue manager configuration.

This example uses the multi-instance configuration described in "Create a multi-instance queue manager on domain controllers" on page 1087. Use the configuration to verify that the multi-instance queue manager switches over to the standby instance. Stop the queue manager with the **endmqm** command and use the -s, switchover, option. The client programs reconnect to the new queue manager instance and continue to work with the new instance after a slight delay.

The client is installed in a 400 MB VMware image that is running Windows XP Service Pack 2. For security reasons, it is connected on the same VMware host-only network as the domain servers that are running the multi-instance queue manager. It is sharing the /MQHA folder, which contains the client connection table, to simplify configuration.

Verifying failover using WebSphere MQ Explorer

Before using the sample applications to verify failover, run the WebSphere MQ Explorer on each server. Add both queue manager instances to each explorer using the **Add Remote Queue Manager > Connect directly to a multi-instance queue manager** wizard. Ensure that both instances are running, permitting standby. Close the window running the VMware image with the active instance, virtually powering off the server, or stop the active instance, allowing switchover to standby instance and reconnectable clients to reconnect.

Note: If you power off the server, make sure that it is not the one hosting the MQHA folder!

Note: The **Allow switchover to a standby instance** option might not be available on the Stop Queue Manager dialog. The option is missing because the queue manager is running as a single instance queue manager. You must have started it without the **Permit a standby instance** option. If your request to stop the queue manager is rejected, look at the Details window, possibly there is no standby instance running.

Verifying failover using the sample programs

Choose a server to run the active instance

You might have chosen one of the servers to host the MQHA directory or file system. If you plan to test failover by closing the VMware window running the active server, make sure that it is not the one hosting MQHA!

On the server running the active queue manager instance

1. Modify *ipaddr1* and *ipaddr2* and save the following commands in N:\hasample.tst.

   ```
   DEFINE QLOCAL(SOURCE) REPLACE
   DEFINE QLOCAL(TARGET) REPLACE
   DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN)   TRPTYPE(TCP) +
          MCAUSER(' ') REPLACE
   DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN)  TRPTYPE(TCP) +
          CONNAME('ipaddr1(1414),ipaddr2(1414)') QMNAME(QM1) REPLACE
   START  CHANNEL(CHANNEL1)
   DEFINE LISTENER(LISTENER.TCP) TRPTYPE(TCP) CONTROL(QMGR)
   DISPLAY LISTENER(LISTENER.TCP) CONTROL
   DISPLAY LSSTATUS(LISTENER.TCP) STATUS
   ```

   Note: By leaving the **MCAUSER** parameter blank, the client user ID is sent to the server. The client user ID must have the correct permissions on the servers. An alternative is to set the **MCAUSER** parameter in the SVRCONN channel to the user ID you have configured on the server.
2. Open a command prompt with the path N:\ and run the command:

   ```
   runmqsc -m QM1 < hasample.tst
   ```
3. Verify that the listener is running and has queue manager control, either by inspecting the output of the **runmqsc** command.

   ```
    LISTENER(LISTENER.TCP)CONTROL(QMGR)
    LISTENER(LISTENER.TCP)STATUS(RUNNING)
   ```

   Or, using the WebSphere MQ Explorer that the TCPIP listener is running and has Control = Queue Manager.

On the client

1. Map the shared directory C:\MQHA on the server to N:\ on the client.
2. Open a command prompt with the path N:\ . Set the environment variable MQCHLLIB to point to the client channel definition table (CCDT) on the server:

   ```
   SET MQCHLLIB=N:\data\QM1\@ipcc
   ```
3. At the command prompt type the commands:

   ```
   start amqsghac TARGET QM1
   start amqsmhac -s SOURCE -t TARGET -m QM1
   start amqsphac SOURCE QM1
   ```

   Note: If you have problems, start the applications at a command prompt so that the reason code is printed out on the console, or look at the AMQERR01.LOG file in the N:\data\QM1\errors folder.

On the server running the active queue manager instance

1. Either:
   - Close the window running the VMware image with the active server instance.
   - Using the WebSphere MQ Explorer, stop the active queue manager instance, allowing switchover to the standby instance and instructing reconnectable clients to reconnect.
2. The three clients eventually detect the connection is broken, and then reconnect. In this configuration, if you close the server window, it is taking about seven minutes for all three connections to be re-established. Some connections are reestablished well before others.

Results

```
N:\>amqsphac SOURCE QM1
Sample AMQSPHAC start
target queue is SOURCE
message <Message 1>
message <Message 2>
message <Message 3>
message <Message 4>
message <Message 5>
17:05:25 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:47 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:52 : EVENT : Connection Reconnected
message <Message 6>
message <Message 7>
message <Message 8>
message <Message 9>
```

```
N:\>amqsmhac -s SOURCE -t TARGET -m QM1
Sample AMQSMHA0 start

17:05:25 : EVENT : Connection Reconnecting (Delay: 97ms)
17:05:48 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:53 : EVENT : Connection Reconnected
```

```
N:\>amqsghac TARGET QM1
Sample AMQSGHAC start
message <Message 1>
message <Message 2>
message <Message 3>
message <Message 4>
message <Message 5>
17:05:25 : EVENT : Connection Reconnecting (Delay: 156ms)
17:05:47 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:52 : EVENT : Connection Reconnected
message <Message 6>
message <Message 7>
message <Message 8>
message <Message 9>
```

Secure shared queue manager data and log directories and files on Windows:

This topic describes how you can secure a shared location for queue manager data and log files using a global alternative security group. You can share the location between different instances of a queue manager running on different servers.

Typically you do not set up a shared location for queue manager data and log files. When you install IBM WebSphere MQ for Windows, the installation program creates a home directory of your choosing for any queue managers that are created on that server. It secures the directories with the local mqm group, and configures a user ID for the IBM IBM WebSphere MQ service to access the directories.

When you secure a shared folder with a security group, a user that is permitted to access the folder must have the credentials of the group. Suppose that a folder on a remote file server is secured with the local mqm group on a server called *mars*. Make the user that runs queue manager processes a member of the local mqm group on *mars*. The user has the credentials that match the credentials of the folder on the remote file server. Using those credentials, the queue manager is able to access its data and logs files in the folder. The user that runs queue manager processes on a different server is a member of a different local mqm group which does not have matching credentials. When the queue manager runs on a different server to *mars*, it cannot access the data and log files it created when it ran on *mars* . Even if you make the user a domain user, it has different credentials, because it must acquire the credentials from the local mqm group on *mars*, and it cannot do that from a different server.

Providing the queue manager with a global alternative security group solves the problem; see Figure 173 . Secure a remote folder with a global group. Pass the name of the global group to the queue manager when you create it on *mars*. Pass the global group name as the alternative security group using the **-a[r]** parameter on the **crtmqm** command. If you transfer the queue manager to run on a different server, the name of the security group is transferred with it. The name is transferred in the **AccessMode** stanza in the qm.ini file as a SecurityGroup; for example:

```
AccessMode:
   SecurityGroup=wmq\wmq
```

The **AccessMode** stanza in the qm.ini also includes the RemoveMQMAccess; for example:

```
AccessMode:
   RemoveMQMAccess=<true\false>
```

If this attribute is specified with value true, and an access group has also been given, the local mqm group is not granted access to the queue manager data files.

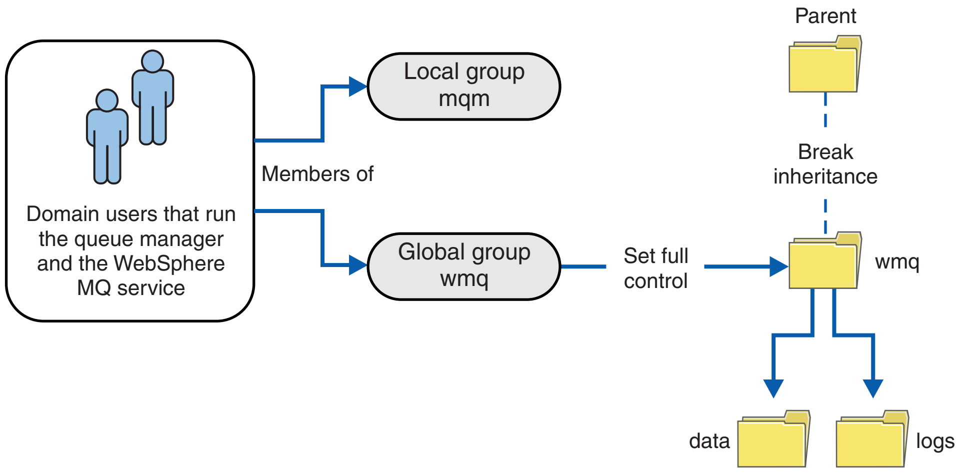


Figure 173. Securing queue manager data and logs using an alternative global security group (1)

For the user ID that queue manager processes are to run with to have the matching credentials of the global security group, the user ID must also have global scope. You cannot make a local group or principal a member of a global group. In Figure 173, the users that run the queue manager processes are shown as domain users.

If you are deploying many IBM WebSphere MQ servers, the grouping of users in Figure 173 is not convenient. You would need to repeat the process of adding users to local groups for every IBM WebSphere MQ server. Instead, create a Domain mqm global group on the domain controller, and make the users that run IBM WebSphere MQ members of the Domain mqm group; see Figure 174 on page 1098 . When you install IBM WebSphere MQ as a domain installation, the “Prepare IBM WebSphere MQ” wizard automatically makes the Domain mqm group a member of the local mqm group. The same users are in both the global groups Domain mqm and wmq.

Tip: The same users can run IBM WebSphere MQ on different servers, but on an individual server you must have different users to run IBM WebSphere MQ as a service, and run interactively. You must also have different users for every installation on a server. Typically, therefore Domain mqm contains a number of users.

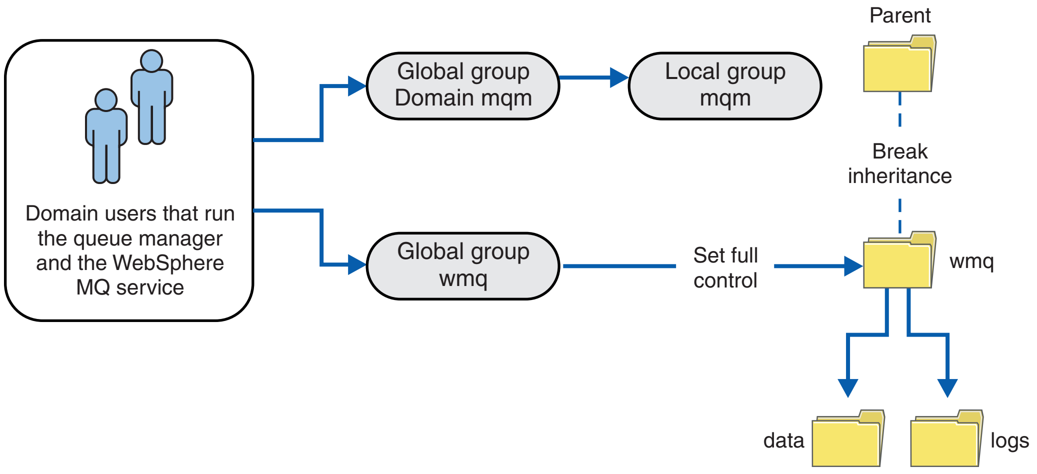


Figure 174. Securing queue manager data and logs using an alternative global security group (2)

The organization in Figure 174 is unnecessarily complicated as it stands. The arrangement has two global groups with identical members. You might simplify the organization, and define only one global group; see Figure 175.

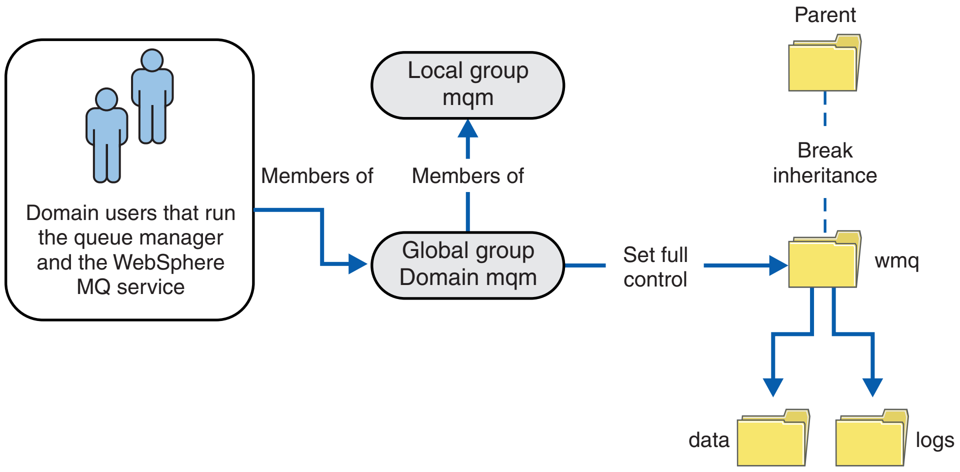


Figure 175. Securing queue manager data and logs using an alternative global security group (3)

Alternatively, you might need a finer degree of access control, with different queue managers restricted to being able to access different folders; see Figure 176 on page 1099. In Figure 176 on page 1099, two groups of domain users are defined, in separate global groups to secure different queue manager log and data files. Two different local `mqm` groups are shown, which must be on different IBM WebSphere MQ servers. In this example, the queue managers are partitioned into two sets, with different users allocated to the two sets. The two sets might be test and production queue managers. The alternate security groups are called `wmq1` and `wmq2` . You must manually add the global groups `wmq1` and `wmq2` to the correct queue managers according to whether they are in the test or production department. The configuration cannot take advantage that the installation of IBM WebSphere MQ propagates `Domain mqm` to the local `mqm` group as in Figure 175, because there are two groups of users.

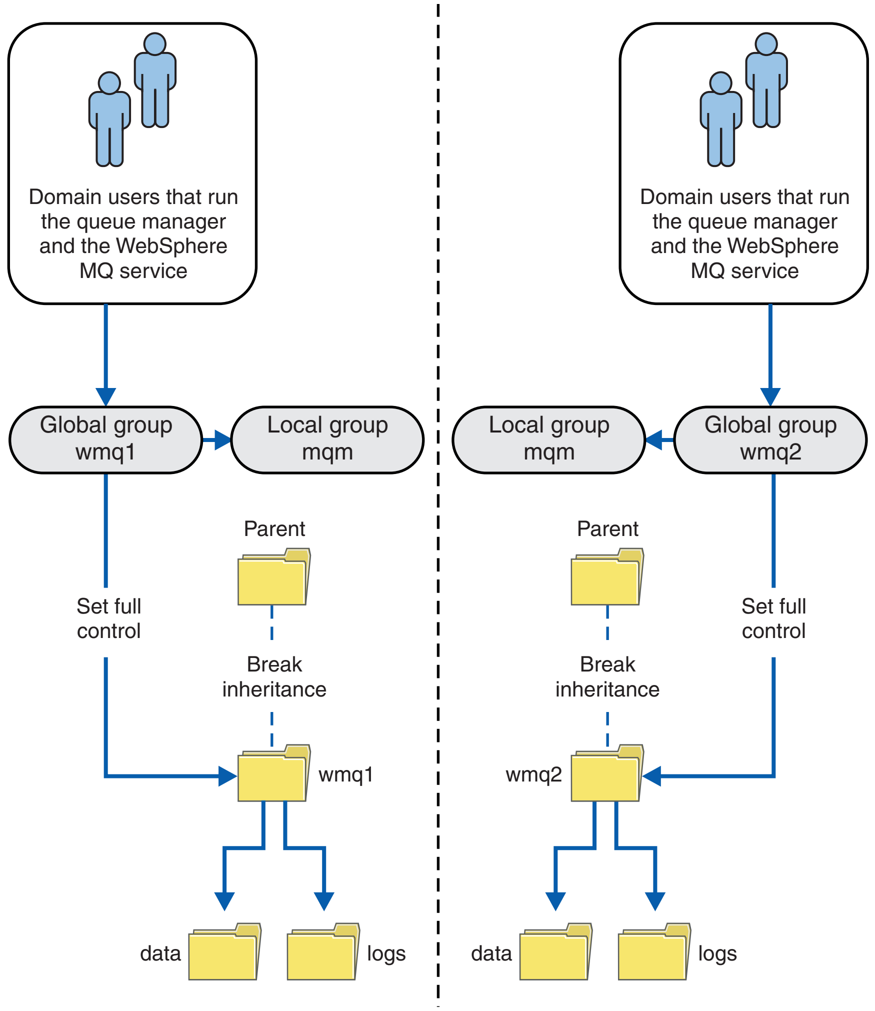


Figure 176. Securing queue manager data and logs using an alternative global security principal (4)

An alternative way to partition two departments would be to place them in two windows domains. In that case, you could return to using the simpler model shown in Figure 175 on page 1098.

Secure unshared queue manager data and log directories and files on Windows:

This topic describes how you can secure an alternative location for queue manager data and log files, both by using the local mqm group and an alternative security group.

Typically you do not set up an alternative location for queue manager data and log files. When you install IBM WebSphere MQ for Windows, the installation program creates a home directory of your choosing for any queue managers that are created. It secures the directories with the local mqm group, and configures a user ID for the IBM IBM WebSphere MQ service to access the directories.

Two examples demonstrate how to configure access control for IBM WebSphere MQ. The examples show how to create a queue manager with its data and logs in directories that are not on the data and log paths created by the installation. In the first example, "Reading and writing data and log files authorized by the local mqm group" on page 1101, you permit access to the queue and log directories by authorizing by the local mqm group. The second example, "Reading and writing data and log files authorized by an alternative local security group" on page 1104, differs in that access to the directories is authorized by an alternative security group. When the directories are accessed by a queue manager running on only one server, securing the data and log files with the alternative security group gives you the choice of securing different queue managers with different local groups or principals. When the directories are accessed by a queue manager running on different servers, such as with a multi-instance queue manager, securing the data and log files with the alternate security group is the only choice; see “Secure shared queue manager data and log directories and files on Windows” on page 1096.

Configuring the security permissions of queue manager data and log files is not a common task on Windows. When you install IBM WebSphere MQ for Windows, you either specify directories for queue manager data and logs, or accept the default directories. The installation program automatically secures these directories with the local mqm group, giving it full control permission. The installation process makes sure the user ID that runs queue managers is a member of the local mqm group. You can modify the other access permissions on the directories to meet your access requirements.

If you move the data and log files directory to new locations, you must configure the security of the new locations. You might change the location of the directories if you back up a queue manager and restore it onto a different computer, or if you change the queue manager to be a multi-instance queue manager. You have a choice of two ways of securing the queue manager data and log directories in their new location. You can secure the directories by restricting access to the local mqm group, or you can restrict access to any security group of your choosing.

It takes the least number of steps to secure the directories using the local mqm group. Set the permissions on the data and log directories to allow the local mqm group full control. A typical approach is to copy the existing set of permissions, removing inheritance from the parent. You can then remove or restrict the permissions of other principals.

If you run the queue manager under a different user ID to the service set up by the Prepare IBM WebSphere MQ wizard, that user ID must be a member of the local mqm group. The task, “Reading and writing data and log files authorized by the local mqm group” on page 1101, takes you through the steps.

You can also secure queue manager data and log files using an alternative security group. The process of securing the queue manager data and log files with the alternative security group has a number of steps that refer to Figure 177. The local group, wmq, is an example of an alternative security group.

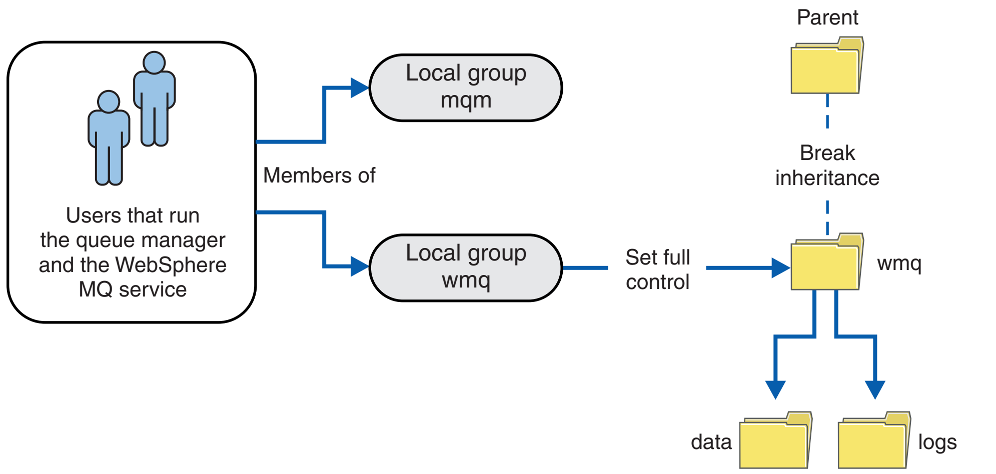


Figure 177. Securing queue manager data and logs using an alternative local security group, wmq

1. Either create separate directories for the queue manager data and logs, a common directory, or a common parent directory.

2. Copy the existing set of inherited permissions for the directories, or parent directory, and modify them according to your needs.
3. Secure the directories that are to contain the queue manager and logs by giving the alternative group, `wmq`, full control permission to the directories.
4. Give all user IDs that run queue manager processes the credentials of the alternative security group or principal:
   a. If you define a user as the alternative security principal, the user must be same user as the queue manager is going to run under. The user must be a member of the local `mqm` group.
   b. If you define a local group as the alternative security group, add the user that the queue manager is going to run under to the alternative group. The user must also be a member of the local `mqm` group.
   c. If you define an global group as the alternative security group, then see "Secure shared queue manager data and log directories and files on Windows" on page 1096.
5. Create the queue manager specifying the alternative security group or principal on the **crtmqm** command, with the **-a** parameter.

Reading and writing data and log files authorized by the local mqm group:

The task illustrates how to create a queue manager with its data and logs files stored in any directory of your choosing. Access to the files is secured by the local `mqm` group. The directory is not shared.

Before you begin

1. Install IBM WebSphere MQ for Windows as the primary installation.
2. Run the "Prepare IBM WebSphere MQ" wizard. For this task, configure the installation either to run with a local user ID, or a domain user ID. Eventually, to complete all the tasks in "Windows domains and multi-instance queue managers" on page 1072, the installation must be configured for a domain.
3. Log on with Administrator authority to perform the first part of the task.

About this task

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, "Windows domains and multi-instance queue managers" on page 1072.

On Windows, you can create the default data and log paths for a IBM WebSphere MQ for Windows in any directories of your choosing. The installation and configuration wizard automatically gives the local `mqm` group, and the user ID that is running the queue manager processes, access to the directories. If you create a queue manager specifying different directories for queue manager data and log files, you must configure full control permission to the directories.

In this example, you give the queue manager full control over its data and log files by giving the local `mqm` group permission to the directory *c:\wmq*.

The **crtmqm** command creates a queue manager that starts automatically when the workstation starts using the IBM IBM WebSphere MQ service.

The task is illustrative; it uses specific values that you can change. The values you can change are in italics. At the end of the task, follow the instructions to remove all the changes you made.

Procedure

1. Open a command prompt.
2. Type the command:

```
md c:\wmq\data , c:\wmq\logs
```

3. Set the permissions on the directories to allow the local mqm group read and write access.

   ```
   cacls c:\wmq /T /E /G mqm:F
   ```

   The system response:

   ```
   processed dir: c:\wmq
   processed dir: c:\wmq\data
   processed dir: c:\wmq\logs
   ```

4. Optional: Switch to a user ID that is a member of the local mqm group.

   You can continue as Administrator, but for a realistic production configuration, continue with a user ID with more restricted rights. The user ID must at least be a member of the local mqm group.

   If the IBM WebSphere MQ installation is configured as part of a domain, make the user ID a member of the Domain mqm group. The "Prepare IBM WebSphere MQ" wizard makes the Domain mqm global group a member of the local mqm group, so you do not have to make the user ID directly a member of the local mqm group.

5. Create the queue manager.

   ```
   crtmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE -md c:\wmq\data -ld c:\wmq\logs  QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager created.
   Directory 'c:\wmq\data\QMGR' created.
   The queue manager is associated with installation '1'
   Creating or replacing default objects for queue manager 'QMGR'
   Default objects statistics : 74 created. 0 replaced.
   Completing setup.
   Setup completed.
   ```

6. Check that the directories created by the queue manager are in the `c:\wmq` directory.

   ```
   dir c:\wmq /D /B /S
   ```

7. Check that the files have read and write, or full control permission for the local mqm group.

   ```
   cacls c:\wmq\*.*
   ```

What to do next

Test the queue manager by putting and getting a message to a queue.

1. Start the queue manager.

   ```
   strmqm QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' starting.
   The queue manager is associated with installation '1'.
   5 log records accessed on queue manager 'QMGR' during the log
   replay phase.
   Log replay for queue manager 'QMGR' complete.
   Transaction manager state recovered for queue manager 'QMGR'.
   WebSphere MQ queue manager 'QMGR' started using V7.1.0.0.
   ```

2. Create a test queue.

   ```
   echo define qlocal(QTEST) | runmqsc QMGR
   ```

   The system response:

```
5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QMGR.


     1 : define qlocal(QTEST)
AMQ8006: WebSphere MQ queue created.
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

3. Put a test message using the sample program **amqsput**.

   ```
   echo 'A test message' | amqsput QTEST QMGR
   ```

   The system response:

   ```
   Sample AMQSPUT0 start
   target queue is QTEST
   Sample AMQSPUT0 end
   ```

4. Get the test message using the sample program **amqsget**.

   ```
   amqsget QTEST QMGR
   ```

   The system response:

   ```
   Sample AMQSGET0 start
   message <A test message>
   Wait 15 seconds ...
   no more messages
   Sample AMQSGET0 end
   ```

5. Stop the queue manager.

   ```
   endmqm -i QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' ending.
   WebSphere MQ queue manager 'QMGR' ended.
   ```

6. Delete the queue manager.

   ```
   dltmqm QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' deleted.
   ```

7. Delete the directories you created.

   Tip: Add the /Q option to the commands to prevent the command prompting to delete each file or directory.

   ```
   del /F /S C:\wmq\*.*
   rmdir /S C:\wmq
   ```

Related concepts:

"Windows domains and multi-instance queue managers" on page 1072
A multi-instance queue manager on Windows requires its data and logs to be shared. The share must be accessible to all instances of the queue manager running on different servers or workstations. Configure the queue managers and share as part of a Windows domain. The queue manager can run on a domain workstation or server, or on the domain controller.

Related tasks:

"Reading and writing data and log files authorized by an alternative local security group"
This task shows how to use the **-a** flag on the **crtmqm** command. The flag provides the queue manager with an alternative local security group to give it access to its log and data files.

"Reading and writing shared data and log files authorized by an alternative global security group" on page 1084
This task shows how to use the **-a** flag on the **crtmqm** command. The **-a** flag gives the queue manager access to its log and data files on a remote file share using the alternative security group.

"Create a multi-instance queue manager on domain workstations or servers" on page 1073
An example shows how to set up a multi-instance queue manager on Windows on a workstation or a server that is part of a Windows domain. The server does not have to be a domain controller. The setup demonstrates the concepts involved, rather than being production scale. The example is based on Windows Server 2008. The steps might differ on other versions of Windows Server.

Reading and writing data and log files authorized by an alternative local security group:

This task shows how to use the **-a** flag on the **crtmqm** command. The flag provides the queue manager with an alternative local security group to give it access to its log and data files.

Before you begin

1. Install IBM WebSphere MQ for Windows as the primary installation.
2. Run the "Prepare IBM WebSphere MQ" wizard. For this task, configure the installation either to run with a local user ID, or a domain user ID. Eventually, to complete all the tasks in "Windows domains and multi-instance queue managers" on page 1072, the installation must be configured for a domain.
3. Log on with Administrator authority to perform the first part of the task.

About this task

This task is one of a set of related tasks that illustrate accessing queue manager data and log files. The tasks show how to create a queue manager authorized to read and write data and log files that are stored in a directory of your choosing. They accompany the task, "Windows domains and multi-instance queue managers" on page 1072.

On Windows, you can create the default data and log paths for a IBM WebSphere MQ for Windows in any directories of your choosing. The installation and configuration wizard automatically gives the local `mqm` group, and the user ID that is running the queue manager processes, access to the directories. If you create a queue manager specifying different directories for queue manager data and log files, you must configure full control permission to the directories.

In this example, you provide the queue manager with an alternative security local group that has full control authorization to the directories. The alternative security group gives the queue manager permission to manage files in the directory. The primary purpose of the alternate security group is to authorize an alternate security global group. Use an alternate security global group to set up a multi-instance queue manager. In this example, you configure a local group to familiarize yourself with the use of an alternate security group without installing IBM WebSphere MQ in a domain. It is unusual to configure a local group as an alternative security group.

The **crtmqm** command creates a queue manager that starts automatically when the workstation starts using the IBM IBM WebSphere MQ service.

The task is illustrative; it uses specific values that you can change. The values you can change are in italics. At the end of the task, follow the instructions to remove all the changes you made.

Procedure

1. Set up an alternative security group.

   The alternative security group is typically a domain group. In the example, you create a queue manager that uses a local alternate security group. With a local alternate security group, you can do the task with a IBM WebSphere MQ installation that is not part of a domain.

   a. Run the **lusrmgr.msc** command to open the Local Users and Groups window.

   b. Right-click **Groups** > **New Group...**

   c. In the **Group name** field, type *altmqm* and click **Create** > **Close**.

   d. Identify the user ID that runs the IBM IBM WebSphere MQ service.

      1) Click **Start** > **Run...**, type services.msc and click **OK**.

      2) Click the IBM IBM WebSphere MQ service in the list of services, and click the Log On tab.

      3) Remember the user ID and close the Services Explorer.

   e. Add the user ID that runs the IBM IBM WebSphere MQ service to the *altmqm* group. Also add the user ID that you log on with to create a queue manager, and run it interactively.

      Windows checks the authority of the queue manager to access the data and logs directories by checking the authority of the user ID that is running queue manager processes. The user ID must be a member, directly or indirectly through a global group, of the *altmqm* group that authorized the directories.

      If you installed IBM WebSphere MQ as part of a domain, and are going to do the tasks in "Create a multi-instance queue manager on domain workstations or servers" on page 1073, the domain user IDs created in "Creating an Active Directory and DNS domain for IBM WebSphere MQ" on page 1076 are *wmquser1* and *wmquser2*.

      If you did not install the queue manager as part of a domain, the default local user ID that runs the IBM IBM WebSphere MQ service is MUSR_MQADMIN. If you intend to do the tasks without Administrator authority, create a user that is a member of the local mqm group.

      Follow these steps to add *wmquser1* and *wmquser2* to *altmqm*. If your configuration is different, substitute your names for the user IDs and group.

      1) In the list of groups right-click **altmqm** > **Properties** > **Add...**.

      2) In the Select Users, Computers, or Groups window type *wmquser1*; *wmquser2* and click **Check Names**.

      3) Type the name and password of a domain administrator in the Windows Security window, then click **OK** > **OK** > **Apply** > **OK**.

2. Open a command prompt.

3. Restart the IBM IBM WebSphere MQ service.

   You must restart the service so that the user ID it runs under acquires the additional security credentials you configured for it.

   Type the commands:

   ```
   endmqsvc
   strmqsvc
   ```

   The system responses:

   ```
   5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
   The MQ service for installation 'Installation1' ended successfully.
   ```

   And:

```
5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
The MQ service for installation 'Installation1' started successfully.
```

4. Type the command:

   ```
   md c:\wmq\data , c:\wmq\logs
   ```

5. Set the permissions on the directories to allow the local user *user* read and write access.

   ```
   cacls c:\wmq /T /E /G altmqm:F
   ```

   The system response:

   ```
   processed dir: c:\wmq
   processed dir: c:\wmq\data
   processed dir: c:\wmq\logs
   ```

6. Optional: Switch to a user ID that is a member of the local mqm group.

   You can continue as Administrator, but for a realistic production configuration, continue with a user ID with more restricted rights. The user ID must at least be a member of the local mqm group.

   If the IBM WebSphere MQ installation is configured as part of a domain, make the user ID a member of the Domain mqm group. The “Prepare IBM WebSphere MQ” wizard makes the Domain mqm global group a member of the local mqm group, so you do not have to make the user ID directly a member of the local mqm group.

7. Create the queue manager.

   ```
   crtmqm -a altmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE -md c:\wmq\data -ld c:\wmq\logs  QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager created.
   Directory 'c:\wmq1\data\QMGR' created.
   The queue manager is associated with installation '1'
   Creating or replacing default objects for queue manager 'QMGR'
   Default objects statistics : 74 created. 0 replaced.
   Completing setup.
   Setup completed.
   ```

8. Check that the directories created by the queue manager are in the *c:\wmq* directory.

   ```
   dir c:\wmq /D /B /S
   ```

9. Check that the files have read and write, or full control permission for the local mqm group.

   ```
   cacls c:\wmq\*.*
   ```

What to do next

Test the queue manager by putting and getting a message to a queue.

1. Start the queue manager.

   ```
   strmqm QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' starting.
   The queue manager is associated with installation '1'.
   5 log records accessed on queue manager 'QMGR' during the log
   replay phase.
   Log replay for queue manager 'QMGR' complete.
   Transaction manager state recovered for queue manager 'QMGR'.
   WebSphere MQ queue manager 'QMGR' started using V7.1.0.0.
   ```

2. Create a test queue.

   ```
   echo define qlocal(QTEST) | runmqsc QMGR
   ```

The system response:

```
5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager QMGR.


     1 : define qlocal(QTEST)
AMQ8006: WebSphere MQ queue created.
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

3. Put a test message using the sample program **amqsput**.

   ```
   echo 'A test message' | amqsput QTEST QMGR
   ```

   The system response:

   ```
   Sample AMQSPUT0 start
   target queue is QTEST
   Sample AMQSPUT0 end
   ```

4. Get the test message using the sample program **amqsget**.

   ```
   amqsget QTEST QMGR
   ```

   The system response:

   ```
   Sample AMQSGET0 start
   message <A test message>
   Wait 15 seconds ...
   no more messages
   Sample AMQSGET0 end
   ```

5. Stop the queue manager.

   ```
   endmqm -i QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' ending.
   WebSphere MQ queue manager 'QMGR' ended.
   ```

6. Delete the queue manager.

   ```
   dltmqm QMGR
   ```

   The system response:

   ```
   WebSphere MQ queue manager 'QMGR' deleted.
   ```

7. Delete the directories you created.

   Tip: Add the /Q option to the commands to prevent the command prompting to delete each file or directory.

   ```
   del /F /S C:\wmq\*.*
   rmdir /S C:\wmq
   ```

Create a multi-instance queue manager on Linux:

An example shows how to set up a multi-instance queue manager on Linux. The setup is small to illustrate the concepts involved. The example is based on Linux Red Hat Enterprise 5. The steps differ on other UNIX platforms.

The example is set up on a 2 GHz notebook computer with 3 GB RAM running Windows XP Service Pack 2. Two VMware virtual machines, Server1 and Server2, run Linux Red Hat Enterprise 5 in 640 MB images. Server1 hosts the network file system (NFS), the queue manager logs and an HA instance. It is not usual practice for the NFS server also to host one of the queue manager instances; this is to simplify the example. Server2 mounts Server1's queue manager logs with a standby instance. A WebSphere MQ MQI client is installed on an additional 400 MB VMware image that runs Windows XP Service Pack 2 and runs the sample high availability applications. All the virtual machines are configured as part of a VMware host-only network for security reasons.

Note: You should put only queue manager data on an NFS server. On the NFS, use the following three options with the mount command to make the system secure:

noexec
By using this option, you stop binary files from being run on the NFS, which prevents a remote user from running unwanted code on the system.

nosuid
By using this option, you prevent the use of the set-user-identifier and set-group-identifier bits, which prevents a remote user from gaining higher privileges.

nodev
By using this option, you stop character and block special devices from being used or defined, which prevents a remote user from getting out of a chroot jail.

Example

Table 131. Illustrative multi-instance queue manager configuration on Linux

Server1	Server2
Log in as *root*	
Follow the instructions in Installing IBM WebSphere MQ to install WebSphere MQ, create the mqm user and group if these do not exist, and define `/var/mqm`. Check what `uid` and `gid` in `/etc/passwd` on the first machine displays for mqm; for example, `mqm:x:501:100:MQ User:/var/mqm:/bin/bash` Match the `uid` and `gid` for mqm in `/etc/passwd` on the second machine to ensure that these values are identical. Reboot this machine if you have to change the values.	
Complete the task Verifying shared file system behavior to check that the file system supports multi-instance queue managers.	
Create log and data directories in a common folder, `/MQHA`, that is to be shared. For example: 1. **`mkdir`** *`/MQHA`* 2. **`mkdir`** *`/MQHA/logs`* 3. **`mkdir`** *`/MQHA/qmgrs`*	Create the folder, `/MQHA`, to mount the shared file system. Keep the path the same as on Server1; for example: 1. **`mkdir`** *`/MQHA`*
Ensure that the `MQHA` directories are owned by user and group mqm, and the access permissions are set to `rwx` for user and group; for example **`ls -al`** displays, `drwxrwxr-x mqm mqm 4096 Nov 27 14:38 MQDATA` 1. **`chown -R`** `mqm:mqm` *`/MQHA`* 2. **`chmod -R`** `ug+rwx` *`/MQHA`*	

Table 131. Illustrative multi-instance queue manager configuration on Linux (continued)

Server1	Server2
Create the queue manager: **`crtmqm -ld`** *`/MQHA/logs`* **`-md`** *`/MQHA/qmgrs QM1`*	
Add(The `'*'` allows all machines that can reach this one mount /MQHA for read/write. Restrict access on a production machine.) `/MQHA *(rw,sync,no_wdelay,fsid=0)` to `/etc/exports`	
Start the NFS daemon: *`/etc/init.d/`*`**nfs** start`	Mount the exported file system `/MQHA`: **`mount -t`** `nfs4` **`-o`** `hard,intr` *`Server1:/ /MQHA`*

Copy the queue manager configuration details from Server1:
`dspmqinf -o` `command QM1`
and copy the result to the clipboard:
`addmqinf -s QueueManager`
` -v Name=QM1`
` -v Directory=QM1`
` -v Prefix=/var/mqm`
` -v DataPath=/MQHA/qmgrs/QM1`
	Paste the queue manager configuration command into Server2:
	`addmqinf -s` `QueueManager`
	`-v` `Name=QM1`
	`-v` `Directory=QM1`
	`-v` `Prefix=/var/mqm`
	`-v` `DataPath=/MQHA/qmgrs/QM1`

Start the queue manager instances, in either order, with the `-x` parameter: **`strmqm -x`** `QM1` The command used to start the queue manager instances must be issued from the same IBM WebSphere MQ installation as the **`addmqinf`** command. To start and stop the queue manager from a different installation, you must first set the installation associated with the queue manager using the **`setmqm`** command. For more information, see setmqm.	

Verifying the multi-instance queue manager on Linux:

Use the sample programs **amqsghac**, **amqsphac** and **amqsmhac** to verify a multi-instance queue manager configuration. This topic provides an example configuration to verify a multi-instance queue manager configuration on Linux Red Hat Enterprise 5.

The high availability sample programs use automatic client reconnection. When the connected queue manager fails, the client attempts to reconnect to a queue manager in the same queue manager group. The description of the samples, High availability sample programs, demonstrates client reconnection using a single instance queue manager for simplicity. You can use the same samples with multi-instance queue managers to verify a multi-instance queue manager configuration.

The example uses the multi-instance configuration described in "Create a multi-instance queue manager on Linux" on page 1108. Use the configuration to verify that the multi-instance queue manager switches over to the standby instance. Stop the queue manager with the **endmqm** command and use the `-s`, switchover, option. The client programs reconnect to the new queue manager instance and continue to work with the new instance after a slight delay.

In the example, the client is running on a Windows XP Service Pack 2 system. The system is hosting two VMware Linux servers that are running the multi-instance queue manager.

Verifying failover using WebSphere MQ Explorer

Before using the sample applications to verify failover, run the WebSphere MQ Explorer on each server. Add both queue manager instances to each explorer using the **Add Remote Queue Manager > Connect directly to a multi-instance queue manager** wizard. Ensure that both instances are running, permitting standby. Close the window running the VMware image with the active instance, virtually powering off the server, or stop the active instance, allowing switchover to standby instance.

Note: If you power off the server, make sure that it is not the one hosting `/MQHA`!

Note: The **Allow switchover to a standby instance** option might not be available on the Stop Queue Manager dialog. The option is missing because the queue manager is running as a single instance queue manager. You must have started it without the **Permit a standby instance** option. If your request to stop the queue manager is rejected, look at the Details window, it is possibly because there is no standby instance running.

Verifying failover using the sample programs

Choose a server to be to run the active instance

You might have chosen one of the servers to host the `MQHA` directory or file system. If you plan to test failover by closing the VMware window running the active server, make sure that it is not the one hosting `MQHA`!

On the server running the active queue manager instance

Note: Running the `SVRCONN` channel with the `MCAUSER` set to `mqm`, is a convenience to reduce the number of configuration steps in the example. If another user ID is chosen, and your system is set up differently to the one used in the example, you might experience access permission problems. Do not use `mqm` as a `MCAUSER` on an exposed system; it is likely to compromise security greatly.

1. Modify *ipaddr1* and *ipaddr2* and save the following commands in `/MQHA/hasamples.tst.`

```
DEFINE  QLOCAL(SOURCE) REPLACE
DEFINE  QLOCAL(TARGET) REPLACE
DEFINE  CHANNEL(CHANNEL1) CHLTYPE(SVRCONN)   TRPTYPE(TCP) +
        MCAUSER('mqm') REPLACE
DEFINE  CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN)  TRPTYPE(TCP) +
        CONNAME('ipaddr1(1414),ipaddr2
(1414)') QMNAME(QM1) REPLACE
START   CHANNEL(CHANNEL1)
DEFINE  LISTENER(LISTENER.TCP) TRPTYPE(TCP) CONTROL(QMGR)
DISPLAY LISTENER(LISTENER.TCP) CONTROL
START   LISTENER(LISTENER.TCP)
DISPLAY LSSTATUS(LISTENER.TCP) STATUS
```

2. Open a terminal window with the path `/MQHA` and run the command:

```
runmqsc -m QM1 < hasamples.tst
```

3. Verify that the listener is running and has queue manager control, either by inspecting the output of the **runmqsc** command.

```
 LISTENER(LISTENER.TCP)CONTROL(QMGR)
 LISTENER(LISTENER.TCP)STATUS(RUNNING)
```

Or, using the WebSphere MQ Explorer that the TCPIP listener is running and has `Control = Queue Manager`.

On the client

1. Copy the client connection table AMQCLCHL.TAB from `/MQHA/qmgrs/QM1.000/@ipcc` on the server to `C:\` on the client.
2. Open a command prompt with the path `C:\` and set the environment variable `MQCHLLIB` to point to the client channel definition table (CCDT)

```
SET MQCHLLIB=C:\
```

3. At the command prompt type the commands:

```
start amqsghac TARGET QM1
start amqsmhac -s SOURCE -t TARGET -m QM1
start amqsphac SOURCE QM1
```

On the server running the active queue manager instance

1. Either:
   - Close the window running the VMware image with the active server instance.
   - Using the WebSphere MQ Explorer, stop the active queue manager instance, allowing switchover to the standby instance and instructing reconnectable clients to reconnect.
2. The three clients eventually detect the connection is broken, and then reconnect. In this configuration, if you close the server window, it is taking about seven minutes for all three connections to be re-established. Some connections are reestablished well before others.

Results

```
N:\>amqsphac SOURCE QM1
Sample AMQSPHAC start
target queue is SOURCE
message <Message 1>
message <Message 2>
message <Message 3>
message <Message 4>
message <Message 5>
17:05:25 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:47 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:52 : EVENT : Connection Reconnected
message <Message 6>
message <Message 7>
message <Message 8>
message <Message 9>
```

```
N:\>amqsmhac -s SOURCE -t TARGET -m QM1
Sample AMQSMHA0 start

17:05:25 : EVENT : Connection Reconnecting (Delay: 97ms)
17:05:48 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:53 : EVENT : Connection Reconnected
```

```
N:\>amqsghac TARGET QM1
Sample AMQSGHAC start
message <Message 1>
message <Message 2>
message <Message 3>
message <Message 4>
message <Message 5>
17:05:25 : EVENT : Connection Reconnecting (Delay: 156ms)
17:05:47 : EVENT : Connection Reconnecting (Delay: 0ms)
17:05:52 : EVENT : Connection Reconnected
message <Message 6>
message <Message 7>
message <Message 8>
message <Message 9>
```

Deleting a multi-instance queue manager:

To delete a multi-instance queue manager completely, you need to use the **dltmqm** command to delete the queue manager, and then remove instances from other servers using either the **rmvmqinf** or **dltmqm** commands.

Run the **dltmqm** command to delete a queue manager that has instances defined on other servers, on any server where that queue manager is defined. You do not need to run the **dltmqm** command on the same server that you created it on. Then run the **rmvmqinf** or **dltmqm** command on all the other servers which have a definition of the queue manager.

You can only delete a queue manager when it is stopped. At the time you delete it no instances are running, and the queue manager, strictly speaking, is neither a single or a multi-instance queue manager; it is simply a queue manager that has its queue manager data and logs on a remote share. When you delete a queue manager, its queue manager data and logs are deleted, and the queue manager stanza is removed from the `mqs.ini` file on the server on which you issued the **dltmqm** command. You need to have access to the network share containing the queue manager data and logs when you delete the queue manager.

On other servers where you have previously created instances of the queue manager there are also entries in the `mqs.ini` files on those servers. You need to visit each server in turn, and remove the queue manager stanza by running the command **rmvmqinf** *Queue manager stanza name*.

On UNIX and Linux systems, if you have placed a common `mqs.ini` file in network storage and referenced it from all the servers by setting the `AMQ_MQS_INI_LOCATION` environment variable on each server, then you need to delete the queue manager from only one of its servers as there is only one `mqs.ini` file to update.

Example

First server

dltmqm QM1

Other servers where instances are defined

rmvmqinf QM1, or

dltmqm QM1

Starting and stopping a multi-instance queue manager:

Starting and stopping a queue manager configured either as a single instance or a multi-instance queue manager.

When you have defined a multi-instance queue manager on a pair of servers, you can run the queue manager on either server, either as a single instance queue manager, or as a multi-instance queue manager.

To run a multi-instance queue manager, start the queue manager on one of the servers using the **strmqm** `-x` *QM1* command; the `-x` option permits the instance to failover. It becomes the *active instance*. Start the standby instance on the other server using the same **strmqm** `-x` *QM1* command; the `-x` option permits the instance to start as a standby.

The queue manager is now running with one active instance that is processing all requests, and a standby instance that is ready to take over if the active instance fails. The active instance is granted exclusive access to the queue manager data and logs. The standby waits to be granted exclusive access to the queue manager data and logs. When the standby is granted exclusive access, it becomes the active instance.

You can also manually switch control to the standby instance by issuing the **endmqm** -s command on the active instance. The **endmqm** -s command shuts down the active instance without shutting down the standby. The exclusive access lock on the queue manager data and logs is released, and the standby takes over.

You can also start and stop a queue manager configured with multiple instances on different servers as a single instance queue manager. If you start the queue manager without using the -x option on the **strmqm** command, the instances of the queue manager configured on other machines are prevented from starting as standby instances. If you attempt to start another instance you receive the response that the queue manager instance is not permitted to run as a standby.

If you stop the active instance of a multi-instance queue manager using the **endmqm** command without the -s option, then the active and standby instances both stop. If you stop the standby instance using the **endmqm** command with the -x option, then it stops being a standby, and the active instance continues running. You cannot issue **endmqm** without the -x option on the standby.

Only two queue manager instances can run at the same time; one is the active instance, and the other is a standby instance. If you start two instances at the same time, WebSphere MQ has no control over which instance becomes the active instance; it is determined by the network file system. The first instance to acquire exclusive access to the queue manager data becomes the active instance.

Note: Before you restart a failed queue manager, you must disconnect your applications from that instance of the queue manager. If you do not, the queue manager might not restart correctly.

Shared file system:

A multi-instance queue manager uses a networked file system to manage queue manager instances.

A multi-instance queue manager automates failover using a combination of file system locks and shared queue manager data and logs. Only one instance of a queue manager can have exclusive access to the shared queue manager data and logs. When it gets access it becomes the active instance. The other instance that does not succeed in getting exclusive access waits as a standby instance until the queue manager data and logs become available.

The networked file system is responsible for releasing the locks it holds for the active queue manager instance. If the active instance fails in some way, the networked file system releases the locks it is holding for the active instance. As soon as the exclusive lock is released, a standby queue manager waiting for the lock attempts to acquire it. If it succeeds, it becomes the active instance and has exclusive access to the queue manager data and logs on the shared file system. It then continues to start.

The related topic, Planning file system support describes how to set up and check that your file system supports multi-instance queue managers.

A multi-instance queue manager does not protect you against a failure in the file system. There are a number of ways to protect your data.

- Invest in reliable storage, such as redundant disk arrays (RAID), and include them in a networked file system that has network resilience.
- Back up WebSphere MQ linear logs to alternative media, and if your primary log media fails, recover using the logs on the alternative media. You can use a backup queue manager to administer this process.

Multiple queue manager instances:

A multi-instance queue manager is resilient because it uses a standby queue manager instance to restore queue manager availability after failure.

Replicating queue manager instances is a very effective way to improve the availability of queue manager processes. Using a simple availability model, purely for illustration: if the reliability of one instance of a queue manager is 99% (over one year, cumulative downtime is 3.65 days) then adding another instance of the queue manager increases the availability to 99.99% (over one year, cumulative downtime of about an hour).

This is too simple a model to give you practical numeric estimates of availability. To model availability realistically, you need to collect statistics for the mean time between failures (MTBF) and the mean time to repair (MTTR), and the probability distribution of time between failures and of repair times.

The term, multi-instance queue manager, refers to the combination of active and standby instances of the queue manager that share the queue manager data and logs. Multi-instance queue managers protect you against the failure of queue manager processes by having one instance of the queue manager active on one server, and another instance of the queue manager on standby on another server, ready to take over automatically should the active instance fail.

Failover or switchover:

A standby queue manager instance takes over from the active instance either on request (switchover), or when the active instance fails (failover).

- *Switchover* takes place when a standby instance starts in response to the **endmqm -s** command being issued to the active queue manager instance. You can specify the **endmqm** parameters `-c`, `-i` or `-p` to control how abruptly the queue manager is stopped.

  Note: Switchover only takes place if a standby queue manager instance is already started. The **endmqm -s** command releases the active queue manager lock and permits switchover: it does not start a standby queue manager instance.
- *Failover* occurs when the lock on queue manager data held by the active instance is released because the instance appeared to stop unexpectedly (that is, without an **endmqm** command being issued).

When the standby instance takes over as the active instance, it writes a message to the queue manager error log.

Reconnectable clients are automatically reconnected when a queue manager fails or switches over. You do not need to include the `-r` flag on the **endmqm** command to request client reconnection. Automatic client reconnect is not supported by WebSphere MQ classes for Java.

If you find that you cannot restart a failed instance, even though failover has occurred and the standby instance has become active, check that applications connected locally to the failed instance have not disconnected from the failed instance. Locally connected applications end or disconnect from a failed queue manager instance to ensure that the failed instance can be restarted. Any locally connected applications using shared bindings (which is the default setting) which hold on to a connection to a failed instance act to prevent the instance from being restarted. If it is not possible to end the locally connected applications, or ensure that they disconnect when the local queue manager instance fails, consider using isolated bindings. Locally connected applications using isolated bindings do not prevent the local queue manager instance from being restarted, even if they do not disconnect.

Channel and client reconnection:

Channel and client reconnection is an essential part of restoring message processing after a standby queue manager instance has become active.

Multi-instance queue manager instances are installed on servers with different network addresses. You need to configure WebSphere MQ channels and clients with connection information for all queue manager instances. When a standby takes over, clients and channels are automatically reconnected to the newly active queue manager instance at the new network address. Automatic client reconnect is not supported by WebSphere MQ classes for Java.

The design is different from the way high availability environments such as HA-CMP work. HA-CMP provides a virtual IP address for the cluster and transfer the address to the active server. WebSphere MQ reconnection does not change or reroute IP addresses. It works by reconnecting using the network addresses you have defined in channel definitions and client connections. As an administrator, you need to define the network addresses in channel definitions and client connections to all instances of any multi-instance queue manager. The best way to configure network addresses to a multi-instance queue manager depends on the connection:

Queue manager channels

The `CONNAME` attribute of channels is a comma-separated list of connection names; for example, `CONNAME('127.0.0.1(1234), 192.0.2.0(4321)')`. The connections are tried in the order specified in the connection list until a connection is successfully established. If no connection is successful, the channel attempts to reconnect.

Cluster channels

Typically, no additional configuration is required to make multi-instance queue managers work in a cluster.

If a queue manager connects to a repository queue manager, the repository discovers the network address of the queue manager. It refers to the `CONNAME` of the `CLUSRCVR` channel at the queue manager. On TCPIP, the queue manager automatically sets the `CONNAME` if you omit it, or configure it to blanks. When a standby instance takes over, its IP address replaces the IP address of the previous active instance as the `CONNAME`.

If it is necessary, you can manually configure `CONNAME` with the list of network addresses of the queue manager instances.

Client connections

Client connections can use connection lists, or queue manager groups to select alternative connections. For more information about client reconnection to a multi-instance queue manager see "Automatic client reconnection" on page 1034. Clients need to be compiled to run with WebSphere MQ Version 7.0.1 client libraries or better. They must be connected to at least a Version 7.0.1 queue manager.

When failover occurs, reconnection takes some time. The standby queue manager has to complete its startup. The clients that were connected to the failed queue manager have to detect the connection failure, and start a new client connection. If a new client connection selects the standby queue manager that has become newly active, then the client is reconnected to the same queue manager.

If the client is in the middle of an MQI call during the reconnection, it must tolerate an extended wait before the call completes.

If the failure takes place during a batch transfer on a message channel, the batch is rolled back and restarted.

Switching over is faster than failing over, and takes only as long as stopping one instance of the queue manager and starting another. For a queue manager with only few log records to replay, at best switchover might take of the order of a few seconds. To estimate how long failover takes, you need to add the time that it takes for the failure to be detected. At best the detection takes of the order of 10 seconds, and might be several minutes, depending on the network and the file system.

Application recovery:

Application recovery is the automated continuation of application processing after failover. Application recovery following failover requires careful design. Some applications need to be aware failover has taken place.

The objective of application recovery is for the application to continue processing with only a short delay. Before continuing with new processing, the application must back out and resubmit the unit of work that it was processing during the failure.

A problem for application recovery is loosing the context that is shared between the WebSphere MQ MQI client and the queue manager, and stored in the queue manager. The WebSphere MQ MQI client restores most of the context, but there are some parts of the context that cannot be reliably restored. The following sections describe some properties of application recovery and how they affect the recovery of applications connected to a multi-instance queue manager.

Transactional messaging

From the perspective of delivering messages, failover does not change the persistent properties of WebSphere MQ messaging. If messages are persistent, and correctly managed within units of work, then messages are not lost during a failover.

From the perspective of transaction processing, transactions are either backed out or committed after failover.

Uncommitted transactions are rolled back. After failover, a reconnectable application receives a `MQRC_BACKED_OUT` reason code to indicate that the transaction has failed, and it needs to roll back the transaction by issuing MQBACK. It then needs to restart the transaction again.

Committed transactions are transactions that have reached the second phase of a two-phase commit, or single phase (message only) transactions that have begun MQCMIT .

If the queue manager is the transaction coordinator and MQCMIT has begun the second phase of its two-phase commit before the failure, the transaction successfully completes. The completion is under the control of the queue manager and continues when the queue manager is running again. In a reconnectable application, the MQCMIT call completes normally.

In a single phase commit, which involves only messages, a transaction that has started commit processing completes normally under the control of the queue manager once it is running again. In a reconnectable application, the MQCMIT completes normally.

Reconnectable clients can use single phase transactions under the control of the queue manager as the transaction coordinator. The extended transactional client does not support reconnection. If reconnection is requested when the transactional client connects, the connection succeeds, but without the ability to be reconnected. The connection behaves as if it is not reconnectable.

Application restart or resume

Failover interrupts an application. After a failure an application can restart from the beginning, or it can resume processing following the interruption. The latter is called *automatic client reconnection*. Automatic client reconnect is not supported by WebSphere MQ classes for Java.

With a WebSphere MQ MQI client application, you can set a connection option to reconnect the client automatically. The options are MQCNO_RECONNECT or MQCNO_RECONNECT_Q_MGR . If no option is set, the client does not try to reconnect automatically and the queue manager failure returns MQRC_CONNECTION_BROKEN to the client. You might design the client to try and start a new connection by issuing a new MQCONN or MQCONNX call.

Server programs have to be restarted; they cannot be automatically reconnected by the queue manager at the point they were processing when the queue manager or server failed. WebSphere MQ server programs are typically not restarted on the standby queue manager instance when a multi-instance queue manager instance fails.

You can automate a WebSphere MQ server program to restart on the standby server in two ways:

1. Package your server application as a queue manager service. It is restarted when the standby queue manager restarts.
2. Write your own failover logic, triggered for example, by the failover log message written by a standby queue manager instance when it starts. The application instance then needs to call MQCONN or MQCONNX after it starts, to create a connection to the queue manager.

Detecting failover

Some applications do need to be aware of failover, others do not. Consider these two examples.

1. A messaging application that gets or receives messages over a messaging channel does not normally require the queue manager at the other end of the channel to be running: it is unlikely to be affected if the queue manager at the other end of the channel restarts on a standby instance.
2. A WebSphere MQ MQI client application processes persistent message input from one queue and puts persistent message responses onto another queue as part of a single unit of work: if it handles an MQRC_BACKED_OUT reason code from MQPUT, MQGET or MQCMIT within sync point by issuing MQBACK and restarting the unit of work, then no messages are lost. Additionally the application does not need to do any special processing to deal with a connection failure.

Suppose however, in the second example, that the application is browsing the queue to select the message to process by using the MQGET option, MQGMO_MSG_UNDER_CURSOR. Reconnection resets the browse cursor, and the MQGET call does not return the correct message. In this example, the application has to be aware failover has occurred. Additionally, before issuing another MQGET for the message under the cursor, the application must restore the browse cursor.

Losing the browse cursor is one example of how the application context changes following reconnection. Other cases are documented in "Recovery of an automatically reconnected client" on page 1118.

You have three alternative design patterns for WebSphere MQ MQI client applications following failover. Only one of them does not need to detect the failover.

No reconnection

In this pattern, the application stops all processing on the current connection when the connection is broken. For the application to continue processing, it must establish a new connection with the queue manager. The application is entirely responsible for transferring any state information it requires to continue processing on the new connection. Existing client applications that reconnect with a queue manager after losing their connection are written in this way.

The client receives a reason code, such as MQRC_CONNECTION_BROKEN, or MQRC_Q_MGR_NOT_AVAILABLE from the next MQI call after the connection is lost. The application must discard all its WebSphere MQ state information, such as queue handles, and issue a new MQCONN or MQCONNX call to establish a new connection, and then reopen the WebSphere MQ objects it needs to process.

The default MQI behavior is for the queue manager connection handle to become unusable after a connection with the queue manager is lost. The default is equivalent to setting the `MQCNO_RECONNECT_DISABLED` option on MQCONNX to prevent application reconnection after failover.

Failover tolerant

Write the application so it is unaffected by failover. Sometimes careful error handling is sufficient to deal with failover.

Reconnection aware

Register an `MQCBT_EVENT_HANDLER` event handler with the queue manager. The event handler is posted with `MQRC_RECONNECTING` when the client starts to try to reconnect to the server, and `MQRC_RECONNECTED` after a successful reconnection. You can then run a routine to reestablish a predictable state so that the client application is able to continue processing.

Recovery of an automatically reconnected client

Failover is an unexpected event, and for an automatically reconnected client to work as designed the consequences of reconnection must be predictable.

A major element of turning an unexpected failure into a predictable and reliable recovery is the use of transactions.

In the previous section, an example, 2 on page 1117, was given of an WebSphere MQ MQI client using a local transaction to coordinate MQGET and MQPUT. The client issues an MQCMIT or MQBACK call in response to a MQRC_BACKED_OUT error and then resubmits the backed out transaction. The queue manager failure causes the transaction to be backed out, and the behavior of the client application ensures no transactions, and no messages, are lost.

Not all program state is managed as part of a transaction, and therefore the consequences of reconnection become harder to understand. You need to know how reconnection changes the state of a WebSphere MQ MQI client in order to design your client application to survive queue manager failover.

You might decide to design your application without any special failover code, handling reconnection errors with the same logic as other errors. Alternatively, you might choose to recognize that reconnection requires special error processing, and register an event handler with WebSphere MQ to run a routine to handle failover. The routine might handle the reconnection processing itself, or set a flag to indicate to the main program thread that when it resumes processing it needs to perform recovery processing.

The WebSphere MQ MQI client environment is aware of failover itself, and restores as much context as it can, following reconnection, by storing some state information in the client, and issuing additional MQI calls on behalf of the client application to restore its WebSphere MQ state. For example, handles to objects that were open at the point of failure are restored, and temporary dynamic queues are opened with the same name. But there are changes that are unavoidable and you need your design to deal with these changes. The changes can be categorized into five kinds:

1. New, or previously undiagnosed errors, are returned from MQI calls until a consistent new context state is restored by the application program.

   An example of receiving a new error is the return code `MQRC_CONTEXT_NOT_AVAILABLE` when trying to pass context after saving context before the reconnection. The context cannot be restored after reconnection because the security context is not passed to an unauthorized client program. To do so would let a malicious application program obtain the security context.

   Typically, applications handle common and predictable errors in a carefully designed way, and relegate uncommon errors to a generic error handler. The error handler might disconnect from WebSphere MQ and reconnect again, or even stop the program altogether. To improve continuity might need to deal with some errors in a different way.
2. Non-persistent messages might be lost.

3. Transactions are rolled back.
4. MQGET or MQPUT calls used outside a sync point might be interrupted with the possible loss of a message.
5. Timing induced errors, due to a prolonged wait in an MQI call.

Some details about lost context are listed in the following section.

- Non-persistent messages are discarded, unless put to a queue with the NPMCLASS(HIGH) option, and the queue manager failure did not interrupt the option of storing non-persistent messages on shutdown.
- A non-durable subscription is lost when a connection is broken. On reconnection, it is re-established. Consider using a durable subscription.
- The get-wait interval is recomputed; if its limit is exceeded it returns MQRC_NO_MSG_AVAILABLE. Similarly, subscription expiry is recomputed to give the same overall expiry time.
- The position of the browse cursor in a queue is lost; it is typically reestablished before the first message.
  - MQGET calls that specify MQGMO_BROWSE_MSG_UNDER_CURSOR or MQGMO_MSG_UNDER_CURSOR, fail with reason code MQRC_NO_MSG_AVAILABLE.
  - Messages locked for browsing are unlocked.
  - Browse marked messages with handle scope are unmarked and can be browsed again.
  - Cooperatively browse marked messages are unmarked in most cases.
- Security context is lost. Attempts to use saved message context, such as putting a message with MQPMO_PASS_ALL_CONTEXT fail with MQRC_CONTEXT_NOT_AVAILABLE.
- Message tokens are lost. MQGET using a message token returns the reason code MQRC_NO_MSG_AVAILABLE .

  Note: *MsgId* and *CorrelId*, as they are part of the message, are preserved with the message during failover, and so MQGET using MsgId or CorrelId work as expected.
- Messages put on a queue under sync point in an uncommitted transaction are no longer available.
- Processing messages in a logical order, or in a message group, results in a return code of MQRC_RECONNECT_INCOMPATIBLE after reconnection.
- An MQI call might return MQRC_RECONNECT_FAILED rather than the more general MQRC_CONNECTION_BROKEN that clients typically receive today.
- Reconnection during an MQPUT call outside sync point returns MQRC_CALL_INTERRUPTED if the WebSphere MQ MQI client does not know if the message was delivered to the queue manager successfully. Reconnection during MQCMIT behaves similarly.
- MQRC_CALL_INTERRUPTED is returned - after a successful reconnect - if the WebSphere MQ MQI client has received no response from the queue manager to indicate the success or failure of
  - the delivery of a persistent message using an MQPUT call outside sync point.
  - the delivery of a persistent message or a message with default persistence using an MQPUT1 call outside sync point.
  - the commit of a transaction using an MQCMIT call. The response is only ever returned after a successful reconnect.
- Channels are restarted as new instances (they might also be different channels), and so no channel exit state is retained.
- Temporary dynamic queues are restored as part of the process of recovering reconnectable clients that had temporary dynamic queues open. No messages on a temporary dynamic queue are restored, but applications that had the queue open, or had remembered the name of the queue, are able to continue processing.

  There is the possibility that if the queue is being used by an application other than the one that created it, that it might not be restored quickly enough to be present when it is next referenced. For example, if a client creates a temporary dynamic queue as a reply-to queue, and a reply message is to be placed on the queue by a channel, the queue might not be recovered in time. In this case, the channel would typically place the reply-to message on the dead letter queue.

If a reconnectable client application opens a temporary dynamic queue by name (because another application has already created it), then when reconnection occurs, the WebSphere MQ MQI client is unable to recreate the temporary dynamic queue because it does not have the model to create it from. In the MQI, only one application can open the temporary dynamic queue by model. Other applications that wish to use the temporary dynamic queue must use MQPUT1, or server bindings, or be able to try the reconnection again if it fails.

Only non-persistent messages might be put to a temporary dynamic queue, and these messages are lost during failover; this loss is true for messages being put to a temporary dynamic queue using MQPUT1 during reconnection. If failover occurs during the MQPUT1, the message might not be put, although the MQPUT1 succeeds. One workaround to this problem is to use permanent dynamic queues. Any server bindings application can open the temporary dynamic queue by name because it is not reconnectable.

Data recovery and high availability:

High availability solutions using multi-instance queue managers must include a mechanism to recover data after a storage failure.

A multi-instance queue manager increases the availability of queue manager processes, but not the availability of other components, such as the file system, that the queue manager uses to store messages, and other information.

One way to make data highly available is to use networked resilient data storage. You can either build your own solution using a networked file system and resilient data storage, or you can buy an integrated solution. If you want to combine resilience with disaster recovery, then asynchronous disk replication, which permits disk replication over tens, or hundreds of kilometers, is available.

You can configure the way different WebSphere MQ directories are mapped to storage media, to make the best use of the media. For *multi-instance* queue managers there is an important distinction between two types of WebSphere MQ directories and files.

Directories that must be shared between the instances of a queue manager.
The information that must be shared between different instances of a queue manager is in two directories: the `qmgrs` and `logs` directories. The directories must be on a shared networked file system. You are advised to use a storage media that provides continuous high availability and excellent performance because the data is constantly changing as messages are created and deleted.

Directories and files that do not *have* to be shared between instances of a queue manager.
Some other directories do not have to be shared between different instances of a queue manager, and are quickly restored by means other than using a mirrored file system.

- WebSphere MQ executable files and the tools directory. Replace by reinstalling or by backing up and restoring from a backed up file archive.
- Configuration information that is modified for the installation as a whole. The configuration information is either managed by WebSphere MQ, such as the `mqs.ini` file on Windows, UNIX and Linux systems, or part of your own configuration management such as **`MQSC`** configuration scripts. Back up and restore using a file archive.
- Installation-wide output such as traces, error logs and FFDC files. The files are stored in the `errors` and `trace` subdirectories in the default data directory. The default data directory on UNIX and Linux systems is `/var/mqm` . On Windows the default data directory is the `WebSphere MQ` installation directory.

You can also use a backup queue manager to take regular media backups of a multi-instance queue manager using linear logging. A backup queue manager does not provide recovery that is as fast as from a mirrored file system, and it does not recover changes since the last backup. The backup queue manager mechanism is more appropriate for use in off-site disaster recovery scenarios than recovering a queue manager after a localized storage failure.

Combining WebSphere MQ Availability solutions:

Applications are using other WebSphere MQ capabilities to improve availability. Multi-instance queue managers complement other high availability capabilities.

WebSphere MQ Clusters increase queue availability

You can increase queue availability by creating multiple definitions of a cluster queue; up to one of every queue on each manager in the cluster.

Suppose a member of the cluster fails and then a new message is sent to a cluster queue. Unless the message *has* to go to the queue manager that has failed, the message is sent to another running queue manager in the cluster that has a definition of the queue.

Although clusters greatly increase availability, there are two related failure scenarios that result in messages getting delayed. Building a cluster with multi-instance queue managers reduces the chance of a message being delayed.

Marooned messages

If a queue manager in the cluster fails, no more messages that can be routed to other queue managers in the cluster are routed to the failed queue manager. Messages that have already been sent are marooned until the failed queue manager is restarted.

Affinities

Affinity is the term used to describe information shared between two otherwise separate computations. For example, an affinity exists between an application sending a request message to a server and the same application expecting to process the reply. Another example would be a sequence of messages, the processing of each message depending on the previous messages.

If you send messages to clustered queues you need to consider affinities. Do you need to send successive messages to the same queue manager, or can each message go to any member of the cluster?

If you do need to send messages to the same queue manager in the cluster and it fails, the messages wait in the transmission queue of the sender until the failed cluster queue manager is running again.

If the cluster is configured with multi-instance queue managers the delay waiting for the failed queue manager to restart is limited to the order of a minute or so while the standby takes over. When the standby is running, marooned messages resume processing, channels to the newly activated queue manager instance are started, and the messages that were waiting in transmission queues start flowing.

A possible way to configure a cluster to overcome messages being delayed by a failed queue manager, is to deploy two different queue managers to each server in the cluster, and arrange for one to be the active and one to be the standby instance of the different queue managers. This is an active-standby configuration, and it increases the availability of the cluster.

As well as having the benefits of reduced administration and increased scalability, clusters continue to provide additional elements of availability to complement multi-instance queue managers. Clusters protect against other types of failure that affect both the active and standby instances of a queue manager.

Uninterrupted service

A cluster provides an uninterrupted service. New messages received by the cluster are sent to active queue managers to be processed. Do not rely on a multi-instance queue manager to provide an uninterrupted service because it takes time for the standby queue manager to detect the failure and complete its startup, for its channels to be reconnected, and for failed batches of messages to be resubmitted.

Localized outage

There are practical limitations to how far apart the active, standby, and file system servers can be, as they need to interact at millisecond speeds to deliver acceptable performance.

Clustered queue managers require interaction speeds of the order of many seconds, and can be geographically dispersed anywhere in the world.

Operational error

By using two different mechanisms to increase availability you reduce the chances that an operational error, such as a human error, compromises your availability efforts.

Queue sharing groups increase message processing availability

Queue sharing groups, provided only on z/OS, allow a group of queue managers to share servicing a queue. If one queue manager fails, the other queue managers continue to process all the messages on the queue. Multi-instance queue managers are not supported on z/OS and complement queue sharing groups only as part of a wider messaging architecture.

WebSphere MQ Clients increase application availability

WebSphere MQ MQI client programs can connect to different queue managers in a queue manager group based on queue manager availability, connection weightings, and affinities. By running an application on a different machine from the one on which the queue manager is running, you can to improve the overall availability of a solution as long as there is a way to reconnect the application if the queue manager instance it is connected to fails.

Queue manager groups are used to increase client availability by uncoupling a client from a queue manager that is stopped, and load balancing client connections across a group of queue managers, rather like an IP sprayer. The client application must have no affinities with the failed queue manager, such as a dependency on a particular queue, or it cannot resume processing.

Automatic client reconnection and multi-instance queue managers increase client availability by resolving some affinity problems. Automatic client reconnect is not supported by WebSphere MQ classes for Java.

You can set the `MQCNO` option `MQCNO_RECONNECT_Q_MGR`, to force a client to reconnect to the same queue manager:

1. If the previously connected single instance queue manager is not running the connection is tried again until the queue manager is running again.
2. If the queue manager is configured as a multi-instance queue manager, then the client reconnects to whichever instance is active.

By automatically reconnecting to the same queue manager, much of the state information the queue manager was holding on behalf of the client, such as the queues it had open and the topic it was subscribed to, are restored. If the client had opened a dynamic reply-to queue to receive a reply to a request, the connection to the reply-to queue is restored too.

Making sure that messages are not lost (logging)

WebSphere MQ logs all the information you need to recover from a queue manager failure.

WebSphere MQ records all significant changes to the data controlled by the queue manager in a recovery log.

This includes creating and deleting objects, persistent message updates, transaction states, changes to object attributes, and channel activities. The log contains the information you need to recover all updates to message queues by:

- Keeping records of queue manager changes
- Keeping records of queue updates for use by the restart process
- Enabling you to restore data after a hardware or software failure

However, WebSphere MQ also relies on the disk system hosting its files. If the disk system is itself unreliable, information, including log information, can still be lost.

What logs look like

Logs consist of primary and secondary files, and a control file. You define the number and size of log files and where they are stored in the file system.

A WebSphere MQ log consists of two components:

1. One or more files of log data.
2. A log control file

A file of log data is also known as a log extent.

There are a number of log files that contain the data being recorded. You can define the number and size (as explained in “Changing IBM WebSphere MQ and queue manager configuration information” on page 1143), or take the system default of three files.

In WebSphere MQ for Windows, each of the three files defaults to 1 MB. In WebSphere MQ for UNIX and Linux systems, each of the three files defaults to 4 MB.

When you create a queue manager, the number of log files you define is the number of *primary* log files allocated. If you do not specify a number, the default value is used.

In WebSphere MQ for Windows, if you have not changed the log path, log files are created under the directory:

```
C:\Program Files\IBM\WebSphere MQ\log\<QMgrName>
```

In WebSphere MQ for UNIX and Linux systems, if you have not changed the log path, log files are created under the directory:

```
/var/mqm/log/<QMgrName>
```

WebSphere MQ starts with these primary log files, but if the primary log space is not sufficient, it allocates *secondary* log files. It does this dynamically and removes them when the demand for log space reduces. By default, up to two secondary log files can be allocated. You can change this default allocation, as described in “Changing IBM WebSphere MQ and queue manager configuration information” on page 1143.

The log control file:

The log control file contains the information needed to control the use of log files, such as their size and location and the name of the next available file.

The log control file is for internal queue manager use only.

The queue manager keeps control data associated with the state of the recovery log in the log control file and you must not modify the contents of the log control file.

Note: Ensure that the logs created when you start a queue manager are large enough to accommodate the size and volume of messages that your applications will handle. You will probably need to change the default log numbers and sizes to meet your requirements. For more information, see “Calculating the size of the log” on page 1127.

Types of logging

In WebSphere MQ, the number of files that are required for logging depends on the file size, the number of messages you have received, and the length of the messages. There are two ways of maintaining records of queue manager activities: circular logging and linear logging.

Circular logging

Use circular logging if all you want is restart recovery, using the log to roll back transactions that were in progress when the system stopped.

Circular logging keeps all restart data in a ring of log files. Logging fills the first file in the ring, then moves on to the next, and so on, until all the files are full. It then goes back to the first file in the ring and starts again. This continues as long as the product is in use, and has the advantage that you never run out of log files.

WebSphere MQ keeps the log entries required to restart the queue manager without loss of data until they are no longer required to ensure queue manager data recovery. The mechanism for releasing log files for reuse is described in “Using checkpointing to ensure complete recovery” on page 1125.

Linear logging

Use linear logging if you want both restart recovery and media recovery (re-creating lost or damaged data by replaying the contents of the log). Linear logging keeps the log data in a continuous sequence of files. Space is not reused, so you can always retrieve any record logged in any log extent that has not been deleted.

As disk space is finite, you might have to think about some form of archiving. It is an administrative task to manage your disk space for the log, reusing or extending the existing space as necessary.

The number of log files used with linear logging can be very large, depending on your message flow and the age of your queue manager. However, there are a number of files that are said to be *active*. Active files contain the log entries required to restart the queue manager. Collectively, active log files are known as the *active log*. The number of active log files is usually less than the number of primary log files as defined in the configuration files. (See “Calculating the size of the log” on page 1127 for information about defining the number.)

The key event that controls whether a log file is termed active or not is a *checkpoint*. A WebSphere MQ checkpoint is a point of consistency between the recovery log and object files. A checkpoint determines the set of log files needed to perform restart recovery. Log files that are not active are not required for restart recovery, and are termed inactive. In some cases inactive log files are required for media recovery. (See "Using checkpointing to ensure complete recovery" for further information about checkpointing.)

Inactive log files can be archived because they are not required for restart recovery. Inactive log files that are not required for media recovery can be considered as superfluous log files. You can delete superfluous log files if they are no longer of interest to your operation. Refer to "Managing logs" on page 1129 for further information about the disposition of log files.

If a new checkpoint is recorded in the second, or later, primary log file, the first file can become inactive and a new primary file is formatted and added to the end of the primary pool, restoring the number of primary files available for logging. In this way the primary log file pool can be seen to be a current set of files in an ever-extending list of log files. Again, it is an administrative task to manage the inactive files according to the requirements of your operation.

Although secondary log files are defined for linear logging, they are not used in normal operation. If a situation arises when, probably due to long-lived transactions, it is not possible to free a file from the active pool because it might still be required for a restart, secondary files are formatted and added to the active log file pool.

If the number of secondary files available is used up, requests for most further operations requiring log activity will be refused with an MQRC_RESOURCE_PROBLEM return code being returned to the application.

Both types of logging can cope with unexpected loss of power, assuming that there is no hardware failure.

Using checkpointing to ensure complete recovery

Checkpoints synchronize queue manager data and log files, and mark a point of consistency from which log records can be discarded. Frequent checkpointing makes recovery quicker.

Persistent updates to message queues happen in two stages. First, the records representing the update are written to the log, then the queue file is updated. The log files can thus become more up to date than the queue files. To ensure that restart processing begins from a consistent point, WebSphere MQ uses checkpoints. A checkpoint is a point in time when the record described in the log is the same as the record in the queue. The checkpoint itself consists of the series of log records needed to restart the queue manager; for example, the state of all transactions (units of work) active at the time of the checkpoint.

WebSphere MQ generates checkpoints automatically. They are taken when the queue manager starts, at shutdown, when logging space is running low, and after every 10 000 operations logged.

As the queues handle further messages, the checkpoint record becomes inconsistent with the current state of the queues.

When WebSphere MQ restarts, it finds the latest checkpoint record in the log. This information is held in the checkpoint file that is updated at the end of every checkpoint. The checkpoint record represents the most recent point of consistency between the log and the data. All the operations that have taken place since the checkpoint are replayed forward. This is known as the replay phase. The replay phase brings the queues back to the logical state they were in before the system failure or shutdown. During the replay phase a list is created of the transactions that were in-flight when the system failure or shutdown occurred. Messages AMQ7229 and AMQ7230 are issued to indicate the progression of the replay phase.

In order to know which operations to back out or commit, WebSphere MQ accesses each active log record associated with an in-flight transaction. This is known as the recovery phase. Messages AMQ7231, AMQ7232 and AMQ7234 are issued to indicate the progression of the recovery phase.

Once all the necessary log records have been accessed during the recovery phase, each active transaction is in turn resolved and each operation associated with the transaction will be either backed out or committed. This is known as the resolution phase. Message AMQ7233 is issued to indicate the progression of the resolution phase.

WebSphere MQ maintains internal pointers to the head and tail of the log. It moves the head pointer to the most recent checkpoint consistent with recovering message data.

Checkpoints are used to make recovery more efficient, and to control the reuse of primary and secondary log files.

In Figure 178, all records before the latest checkpoint, Checkpoint 2, are no longer needed by WebSphere MQ. The queues can be recovered from the checkpoint information and any later log entries. For circular logging, any freed files before the checkpoint can be reused. For a linear log, the freed log files no longer need to be accessed for normal operation and become inactive. In the example, the queue head pointer is moved to point at the latest checkpoint, Checkpoint 2, which then becomes the new queue head, Head 2. Log File 1 can now be reused.

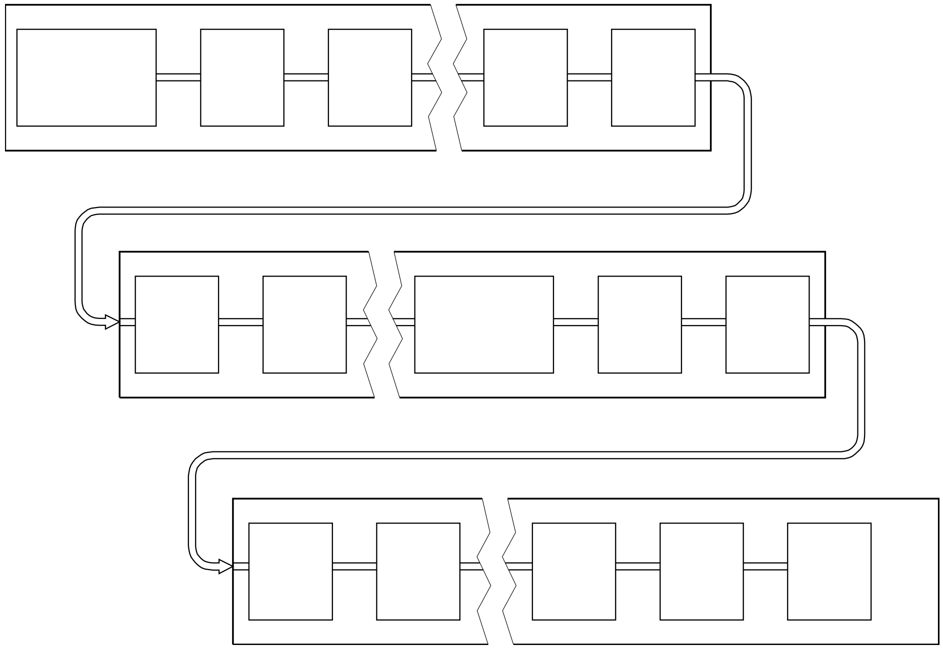


Figure 178. Checkpointing. For simplicity, only the ends of the log files are shown.

Checkpointing with long-running transactions:

How a long-running transaction affects reuse of log files.

Figure 179 on page 1127 shows how a long-running transaction affects reuse of log files. In the example, a long-running transaction has made an entry to the log, shown as LR 1, after the first checkpoint shown. The transaction does not complete (at point LR 2) until after the third checkpoint. All the log information from LR 1 onwards is retained to allow recovery of that transaction, if necessary, until it has completed.

After the long-running transaction has completed, at LR 2, the head of the log is moved to Checkpoint 3, the latest logged checkpoint. The files containing log records before Checkpoint 3, Head 2, are no longer needed. If you are using circular logging, the space can be reused.

If the primary log files are completely full before the long-running transaction completes, secondary log files are used to avoid the logs getting full.

When the log head is moved and you are using circular logging, the primary log files might become eligible for reuse and the logger, after filling the current file, reuses the first primary file available to it. If you are using linear logging, the log head is still moved down the active pool and the first file becomes inactive. A new primary file is formatted and added to the bottom of the pool in readiness for future logging activities.

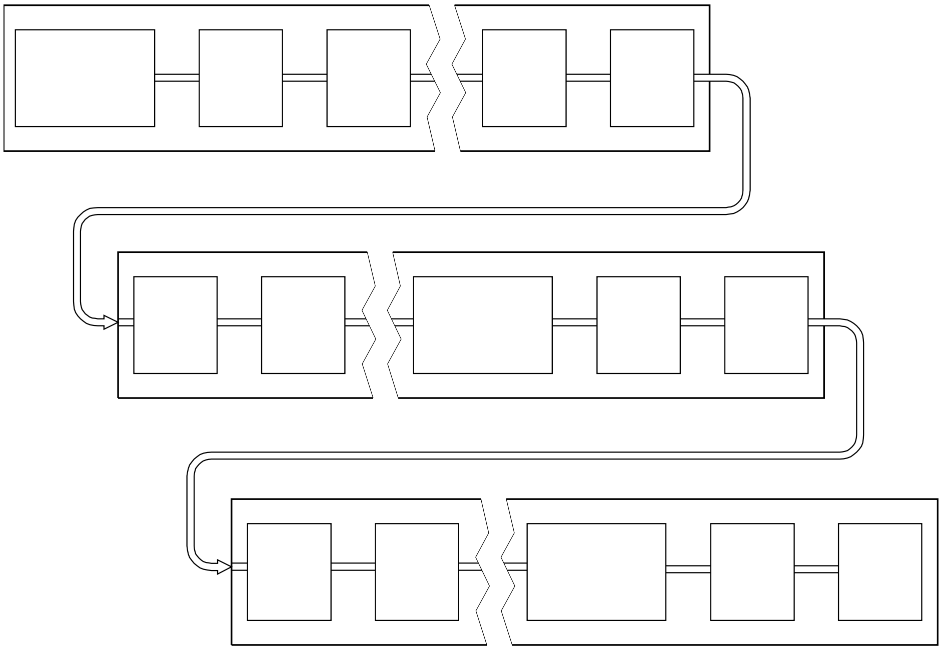


Figure 179. Checkpointing with a long-running transaction. For simplicity, only the ends of the log files are shown.

Calculating the size of the log

Estimating the size of log a queue manager needs.

After deciding whether the queue manager uses circular or linear logging, you need to estimate the size of the log that the queue manager needs. The size of the log is determined by the following log configuration parameters:

LogFilePages
The size of each primary and secondary log file in units of 4K pages

LogPrimaryFiles
The number of preallocated primary log files

LogSecondaryFiles
The number of secondary log files that can be created for use when the primary log files are full

Table 132 on page 1128 shows the amount of data the queue manager logs for various operations. Most queue manager operations need a minimal amount of log space. However, when a persistent message is put to a queue, **all** the message data must be written to the log to make it possible to recover the message. The size of the log depends, typically, on the number and size of the persistent messages the queue manager needs to handle.

Table 132. Log entry sizes (all values are approximate)

Operation	Size
Put persistent message	750 bytes + message length If the message is large, it is divided into segments of 261844 bytes, each segment adding an extra 300 bytes.
Get message	260 bytes
Sync point, commit	750 bytes
Sync point, rollback	1000 bytes + 12 bytes for each get or put to be rolled back
Create object	1500 bytes
Delete object	300 bytes
Alter attributes	1024 bytes
Record media image	800 bytes + image The image is divided into segments of 260 000 bytes, each segment adding an extra 300 bytes.
Checkpoint	750 bytes + 200 bytes for each active unit of work + 380 bytes for each cluster-sender channel, if you are using multiple cluster transmission queues per queue manager. Additional data might be logged for any uncommitted puts or gets that are buffered for performance reasons. If you have cluster-sender channels, then at each checkpoint an extra 380 bytes is written to the log per cluster-sender channel.

Note:

1. You can change the number of primary and secondary log files each time the queue manager starts.
2. You cannot change the log file size; you must determine it **before** creating the queue manager.
3. The number of primary log files and the log file size determine the amount of log space that is preallocated when the queue manager is created.
4. The total number of primary and secondary log files cannot exceed 511 on UNIX and Linux systems, or 255 on Windows, which in the presence of long-running transactions, limits the maximum amount of log space available to the queue manager for restart recovery. The amount of log space the queue manager might need for media recovery does not share this limit.
5. When *circular* logging is being used, the queue manager reuses primary log space. This means that the queue manager's log can be smaller than the amount of data you estimated that the queue manager needs to log. The queue manager will, up to a limit, allocate a secondary log file when a log file becomes full, and the next primary log file in the sequence is not available.
6. Primary log files are made available for reuse during a checkpoint. The queue manager takes both the primary and secondary log space in to consideration before taking a checkpoint because the amount of log space is running low.

   If you do not define more primary log files than secondary log files, the queue manager might allocate secondary log files before a checkpoint is taken. This makes the primary log files available for reuse.

Managing logs

Logs are nearly self-managing, but sometimes need managing to resolve space problems.

Over time, some of the log records written become unnecessary for restarting the queue manager. If you are using circular logging, the queue manager reclaims freed space in the log files. This activity is not apparent to the user and you do not usually see the amount of disk space used reduce because the space allocated is quickly reused.

Of the log records, only those written since the start of the last complete checkpoint, and those written by any active transactions, are needed to restart the queue manager. Thus, the log might fill if a checkpoint has not been taken for a long time, or if a long-running transaction wrote a log record a long time ago. The queue manager tries to take checkpoints often enough to avoid the first problem.

When a long-running transaction fills the log, attempts to write log records fail and some MQI calls return MQRC_RESOURCE_PROBLEM. (Space is reserved to commit or roll back all in-flight transactions, so **MQCMIT** or **MQBACK** should not fail.)

The queue manager rolls back transactions that consume too much log space. An application that has a transaction is rolled back in this way cannot perform subsequent **MQPUT** or **MQGET** operations specifying sync point under the same transaction. An attempt to put or get a message under sync point in this state returns MQRC_BACKED_OUT. The application can then issue **MQCMIT**, which returns MQRC_BACKED_OUT, or **MQBACK** and start a new transaction. When the transaction consuming too much log space has been rolled back, its log space is released and the queue manager continues to operate normally.

If the log fills, message AMQ7463 is issued. In addition, if the log fills because a long-running transaction has prevented the space being released, message AMQ7465 is issued.

Finally, if records are being written to the log faster than the log can process them, message AMQ7466 is issued. If you see this message, increase the number of log files or reduce the amount of data being processed by the queue manager.

What happens when a disk gets full:

The queue manager logging component can cope with a full disk, and with full log files. If the disk containing the log fills, the queue manager issues message AMQ6708 and an error record is taken.

The log files are created at their maximum size, rather than being extended as log records are written to them. This means that WebSphere MQ can run out of disk space only when it is creating a new file; it cannot run out of space when it is writing a record to the log. WebSphere MQ always knows how much space is available in the existing log files, and manages the space within the files accordingly.

If you fill the drive containing the log files, you might be able to free some disk space. If you are using a linear log, there might be some inactive log files in the log directory, and you can copy these files to another drive or device. If you still run out of space, check that the configuration of the log in the queue manager configuration file is correct. You might be able to reduce the number of primary or secondary log files so that the log does not outgrow the available space. You cannot alter the size of the log files for an existing queue manager. The queue manager assumes that all log files are the same size.

Managing log files:

Allocate sufficient space for your log files. For linear logging, you can delete old log files when they are no longer required.

If you are using circular logging, ensure that there is sufficient space to hold the log files when you configure your system (see “Log defaults for WebSphere MQ” on page 1153 and “Queue manager logs” on page 1161). The amount of disk space used by the log does not increase beyond the configured size, including space for secondary files to be created when required.

If you are using a linear log, the log files are added continually as data is logged, and the amount of disk space used increases with time. If the rate of data being logged is high, disk space is used rapidly by new log files.

Over time, the older log files for a linear log are no longer required to restart the queue manager or to perform media recovery of any damaged objects. The following are methods for determining which log files are still required:

Logger event messages
When enabled, logger event messages are generated when queue managers starts writing log records to a new log file. The contents of logger event messages specify the log files that are still required for queue manager restart, and media recovery. For more information about logger event messages, see Logger events

Queue manager status
Running the MQSC command, DISPLAY QMSTATUS, or the PCF command, Inquire Queue Manager Status, returns queue manager information, including details of the required log files. For more information about MQSC commands, see Script (MQSC) Commands , and for information about PCF commands, see Automating administration tasks.

Queue manager messages
Periodically, the queue manager issues a pair of messages to indicate which of the log files are needed:

- Message AMQ7467 gives the name of the oldest log file required to restart the queue manager. This log file and all newer log files must be available during queue manager restart.
- Message AMQ7468 gives the name of the oldest log file needed for media recovery.

Only log files required for queue manager restart, active log files, are required to be online. Inactive log files can be copied to an archive medium such as tape for disaster recovery, and removed from the log directory. Inactive log files that are not required for media recovery can be considered as superfluous log files. You can delete superfluous log files if they are no longer of interest to your operation.

To determine "older" and "newer" log files, use the log file number rather than the modification times applied by the file system.

If any log file that is needed cannot be found, operator message AMQ6767 is issued. Make the log file, and all subsequent log files, available to the queue manager and try the operation again.

Note: When performing media recovery, all the required log files must be available in the log file directory at the same time. Make sure that you take regular media images of any objects you might want to recover to avoid running out of disk space to hold all the required log files. To take a media image of all your objects in your queue manager, run the **rcdmqimg** command as shown in the following examples:

On Windows

```
rcdmqimg -m QMNAME -t all *
```

On UNIX and Linux

```
rcdmqimg -m QMNAME -t all "*"
```

Running **rcdmqimg** moves the media log sequence number (LSN) forwards. For further details on log sequence numbers, see “Dumping the contents of the log using the dmpmqlog command” on page 1134. **rcdmqimg** does not run automatically, therefore must be run manually or from an automatic task you have created. For more information about this command, see rcdmqimg and dmpmqlog.

Note: Messages AMQ7467 and AMQ7468 can also be issued at the time of running the **rcdmqimg** command.

Determining superfluous log files:

When managing linear log files, it is important to be sure which files can be deleted or archived. This information will assist you in making this decision.

Do not use the file system's modification times to determine "older" log files. Use only the log file number. The queue manager's use of log files follows complex rules, including pre-allocation and formatting of log files before they are needed. You might see log files with modification times that would be misleading if you try to use these times to determine relative age.

To determine the oldest log file needed to restart the queue manager, issue the command `DISPLAY QMSTATUS RECLOG`.

To determine the oldest log file needed to perform media recovery, issue the command `DISPLAY QMSTATUS MEDIALOG`.

In general a lower log file number implies an older log. Unless you have a very high log file turnover, of the order of 3000 log files per day for 10 years, then you do not need to cater for the number wrapping at 9 999 999. In this case, you can archive any log file with a number less than the RECLOG value, and you can delete any log file with a number less than both the RECLOG and MEDIALOG values.

If however you have a very high turnover of log files, or otherwise want to be confident of coping with the general case, then the following algorithm can typically be used:

```
Let S == restart log file number
                (from DISPLAY QMSTATUS RECLOG).
Let M == media recovery log file number
                 (from DISPLAY QMSTATUS MEDIALOG).
Let L == a log file number with eligibility for deletion or archiving
                  that needs to be determined.

function minlog (a, b) {
    if (abs (a - b) < 5000000)
        return min (a, b);  # Not wrapped.
    else
        return max (a, b);  # Wrapped. }

A log file L can be deleted if
          (L != S && L != M && minlog (L, minlog (S, M)) == L).
A log file L can be archived if
          (L != S && minlog (L, S) == L).
```

Log file location:

When choosing a location for your log files, remember that operation is severely affected if WebSphere MQ fails to format a new log because of lack of disk space.

If you are using a circular log, ensure that there is sufficient space on the drive for at least the configured primary log files. Also leave space for at least one secondary log file, which is needed if the log has to grow.

If you are using a linear log, allow considerably more space; the space consumed by the log increases continuously as data is logged.

Ideally, place the log files on a separate disk drive from the queue manager data. This has benefits in terms of performance. It might also be possible to place the log files on multiple disk drives in a mirrored arrangement. This protects against failure of the drive containing the log. Without mirroring, you could be forced to go back to the last backup of your WebSphere MQ system.

Using the log for recovery

Using logs to recover from failures.

There are several ways that your data can be damaged. WebSphere MQ helps you to recover from:
- A damaged data object
- A power loss in the system
- A communications failure

This section looks at how the logs are used to recover from these problems.

Recovering from power loss or communications failures:

WebSphere MQ can recover from both communications failures and loss of power. In addition, it can sometimes recover from other types of problem, such as inadvertent deletion of a file.

In the case of a communications failure, messages remain on queues until they are removed by a receiving application. If the message is being transmitted, it remains on the transmission queue until it can be successfully transmitted. To recover from a communications failure, you can usually restart the channels using the link that failed.

If you lose power, when the queue manager is restarted WebSphere MQ restores the queues to their committed state at the time of the failure. This ensures that no persistent messages are lost. Nonpersistent messages are discarded; they do not survive when WebSphere MQ stops abruptly.

Recovering damaged objects:

There are ways in which a IBM WebSphere MQ object can become unusable, for example because of inadvertent damage. You must then recover either your complete system or some part of it. The action required depends on when the damage is detected, whether the log method selected supports media recovery, and which objects are damaged.

Media recovery

Media recovery re-creates objects from information recorded in a linear log. For example, if an object file is inadvertently deleted, or becomes unusable for some other reason, media recovery can re-create it. The information in the log required for media recovery of an object is called a *media image*.

A media image is a sequence of log records containing an image of an object from which the object itself can be re-created.

The first log record required to re-create an object is known as its *media recovery record*; it is the start of the latest media image for the object. The media recovery record of each object is one of the pieces of information recorded during a checkpoint.

When an object is re-created from its media image, it is also necessary to replay any log records describing updates performed on the object since the last image was taken.

Consider, for example, a local queue that has an image of the queue object taken before a persistent message is put onto the queue. In order to re-create the latest image of the object, it is necessary to replay the log entries recording the putting of the message to the queue, in addition to replaying the image itself.

When an object is created, the log records written contain enough information to completely re-create the object. These records make up the first media image of the object. Then, at each shutdown, the queue manager records media images automatically as follows:

- Images of all process objects and queues that are not local
- Images of empty local queues

Media images can also be recorded manually using the **rcdmqimg** command, described in rcdmqimg. This command writes a media image of the IBM WebSphere MQ object. When a media image has been written, only the logs that hold the media image, and all the logs created after this time, are required to re-create damaged objects. The benefit of creating media images depends on such factors as the amount of free storage available, and the speed at which log files are created.

Recovering from media images

IBM WebSphere MQ automatically recovers some objects from their media image if it finds that they are corrupted or damaged. In particular, recovery applies to objects found to be damaged during the normal queue manager startup. If any transaction was incomplete when the queue manager last shut down, any queue affected is also recovered automatically in order to complete the startup operation.

You must recover other objects manually, using the **rcrmqobj** command, which replays the records in the log to re-create the IBM WebSphere MQ object. The object is re-created from its latest image found in the log, together with all applicable log events between the time the image was saved and the time the re-create command was issued. If a IBM WebSphere MQ object becomes damaged, the only valid actions that can be performed are either to delete it or to re-create it by this method. Nonpersistent messages cannot be recovered in this way.

See rcrmqobj for further details of the **rcrmqobj** command.

The log file containing the media recovery record, and all subsequent log files, must be available in the log file directory when attempting media recovery of an object. If a required file cannot be found, operator message AMQ6767 is issued and the media recovery operation fails. If you do not take regular media images of the objects that you want to re-create, you might have insufficient disk space to hold all the log files required to re-create an object.

Recovering damaged objects during startup

If the queue manager discovers a damaged object during startup, the action it takes depends on the type of object and whether the queue manager is configured to support media recovery.

If the queue manager object is damaged, the queue manager cannot start unless it can recover the object. If the queue manager is configured with a linear log, and thus supports media recovery, IBM WebSphere MQ automatically tries to re-create the queue manager object from its media images. If the log method selected does not support media recovery, you can either restore a backup of the queue manager or delete the queue manager.

If any transactions were active when the queue manager stopped, the local queues containing the persistent, uncommitted messages put or got inside these transactions are also required to start the queue manager successfully. If any of these local queues is found to be damaged, and the queue manager supports media recovery, it automatically tries to re-create them from their media images. If any of the queues cannot be recovered, IBM WebSphere MQ cannot start.

If any damaged local queues containing uncommitted messages are discovered during startup processing on a queue manager that does not support media recovery, the queues are marked as damaged objects and the uncommitted messages on them are ignored. This situation is because it is not possible to perform media recovery of damaged objects on such a queue manager and the only action left is to delete them. Message AMQ7472 is issued to report any damage.

Recovering damaged objects at other times

Media recovery of objects is automatic only during startup. At other times, when object damage is detected, operator message AMQ7472 is issued and most operations using the object fail. If the queue manager object is damaged at any time after the queue manager has started, the queue manager performs a pre-emptive shutdown. When an object has been damaged you can delete it or, if the queue manager is using a linear log, attempt to recover it from its media image using the **rcrmqobj** command (see rcrmqobj for further details).

Protecting WebSphere MQ log files

Do not touch the log files when a queue manager is running, recovery might be impossible. Use super user or mqm authority to protect log files against inadvertent modification.

Do not remove the active log files manually when a WebSphere MQ queue manager is running. If a user inadvertently deletes the log files that a queue manager needs to restart, WebSphere MQ **does not** issue any errors and continues to process data *including persistent messages*. The queue manager shuts down normally, but can fail to restart. Recovery of messages then becomes impossible.

Users with the authority to remove logs that are being used by an active queue manager also have authority to delete other important queue manager resources (such as queue files, the object catalog, and WebSphere MQ executable files). They can therefore damage, perhaps through inexperience, a running or dormant queue manager in a way against which WebSphere MQ cannot protect itself.

Exercise caution when conferring super user or mqm authority.

Dumping the contents of the log using the dmpmqlog command

How to use the **dmpmqlog** command to dump the contents of the queue manager log.

Use the **dmpmqlog** command to dump the contents of the queue manager log. By default all active log records are dumped, that is, the command starts dumping from the head of the log (usually the start of the last completed checkpoint).

The log can usually be dumped only when the queue manager is not running. Because the queue manager takes a checkpoint during shutdown, the active portion of the log usually contains a small number of log records. However, you can use the **dmpmqlog** command to dump more log records using one of the following options to change the start position of the dump:

- Start dumping from the *base* of the log. The base of the log is the first log record in the log file that contains the head of the log. The amount of additional data dumped in this case depends on where the head of the log is positioned in the log file. If it is near the start of the log file, only a small amount of additional data is dumped. If the head is near the end of the log file, significantly more data is dumped.

- Specify the start position of the dump as an individual log record. Each log record is identified by a unique *log sequence number (LSN)*. In the case of circular logging, this starting log record cannot be before the base of the log; this restriction does not apply to linear logs. You might need to reinstate inactive log files before running the command. You must specify a valid LSN, taken from previous **dmpmqlog** output, as the start position.

  For example, with linear logging you can specify the `nextlsn` from your last **dmpmqlog** output. The `nextlsn` appears in `Log File Header` and indicates the LSN of the next log record to be written. Use this as a start position to format all log records written since the last time the log was dumped.
- **For linear logs only**, you can instruct **dmpmqlog** to start formatting log records from any given log file extent. In this case, **dmpmqlog** expects to find this log file, and each successive one, in the same directory as the active log files. This option does not apply to circular logs, where **dmpmqlog** cannot access log records prior to the base of the log.

The output from the **dmpmqlog** command is the `Log File Header` and a series of formatted log records. The queue manager uses several log records to record changes to its data.

Some of the information that is formatted is only of use internally. The following list includes the most useful log records:

`Log File Header`

Each log has a single log file header, which is always the first thing formatted by the **dmpmqlog** command. It contains the following fields:

logactive	The number of primary log extents.
loginactive	The number of secondary log extents.
logsize	The number of 4 KB pages per extent.
baselsn	The first LSN in the log extent containing the head of the log.
nextlsn	The LSN of the next log record to be written.
headlsn	The LSN of the log record at the head of the log.
tailsn	The LSN identifying the tail position of the log.
hflag1	Whether the log is CIRCULAR or LOG RETAIN (linear).
HeadExtentID	The log extent containing the head of the log.

`Log Record Header`

Each log record within the log has a fixed header containing the following information:

LSN	The log sequence number.
LogRecdType	The type of the log record.
XTranid	The transaction identifier associated with this log record (if any). A *`TranType`* of MQI indicates a WebSphere MQ-only transaction. A *`TranType`* of XA is involved with other resource managers. Updates involved within the same unit of work have the same *`XTranid`*.
QueueName	The queue associated with this log record (if any).
Qid	The unique internal identifier for the queue.
PrevLSN	The LSN of the previous log record within the same transaction (if any).

`Start Queue Manager`

This logs that the queue manager has started.

StartDate	The date that the queue manager started.
StartTime	The time that the queue manager started.

Stop Queue Manager

This logs that the queue manager has stopped.

StopDate	The date that the queue manager stopped.
StopTime	The time that the queue manager stopped.
ForceFlag	The type of shutdown used.

Start Checkpoint

This denotes the start of a queue manager checkpoint.

End Checkpoint

This denotes the end of a queue manager checkpoint.

ChkPtLSN	The LSN of the log record that started this checkpoint.

Put Message

This logs a persistent message put to a queue. If the message was put under sync point, the log record header contains a non-null *XTranid*. The remainder of the record contains:

MapIndex
An identifier for the message on the queue. It can be used to match the corresponding **MQGET** that was used to get this message from the queue. In this case a subsequent *Get Message* log record can be found containing the same QueueName and MapIndex. At this point the MapIndex identifier can be reused for a subsequent put message to that queue.

Data
Contained in the hex dump for this log record is various internal data followed by the Message Descriptor (eyecatcher MD) and the message data itself.

Put Part

Persistent messages that are too large for a single log record are logged as multiple *Put Part* log records followed by a single *Put Message* record. If there are *Put Part* records, then the *PrevLSN* field will chain the *Put Part* records and the final *Put Message* record together.

Data	Continues the message data where the previous log record left off.

Get Message

Only gets of persistent messages are logged. If the message was got under sync point, the log record header contains a non-null *XTranid*. The remainder of the record contains:

MapIndex
Identifies the message that was retrieved from the queue. The most recent *Put Message* log record containing the same *QueueName* and *MapIndex* identifies the message that was retrieved.

QPriority
The priority of the message retrieved from the queue.

Start Transaction

Indicates the start of a new transaction. A TranType of MQI indicates a WebSphere MQ-only transaction. A TranType of XA indicates one that involves other resource managers. All updates made by this transaction will have the same *XTranid*.

Prepare Transaction

Indicates that the queue manager is prepared to commit the updates associated with the specified *XTranid*. This log record is written as part of a two-phase commit involving other resource managers.

`Commit Transaction`
Indicates that the queue manager has committed all updates made by a transaction.

`Rollback Transaction`
This denotes the queue manager's intention to roll back a transaction.

`End Transaction`
This denotes the end of a rolled-back transaction.

`Transaction Table`
This record is written during sync point. It records the state of each transaction that has made persistent updates. For each transaction the following information is recorded:

XTranid	The transaction identifier.
FirstLSN	The LSN of the first log record associated with the transaction.
LastLSN	The LSN of the last log record associated with the transaction.

`Transaction Participants`
This log record is written by the XA Transaction Manager component of the queue manager. It records the external resource managers that are participating in transactions. For each participant the following is recorded:

RMName	The name of the resource manager.
RMID	The resource manager identifier. This is also logged in subsequent *`Transaction Prepared`* log records that record global transactions in which the resource manager is participating.
SwitchFile	The switch load file for this resource manager.
XAOpenString	The XA open string for this resource manager.
XACloseString	The XA close string for this resource manager.

`Transaction Prepared`
This log record is written by the XA Transaction Manager component of the queue manager. It indicates that the specified global transaction has been successfully prepared. Each of the participating resource managers will be instructed to commit. The *`RMID`* of each prepared resource manager is recorded in the log record. If the queue manager itself is participating in the transaction a *`Participant Entry`* with an *`RMID`* of zero will be present.

`Transaction Forget`
This log record is written by the XA Transaction Manager component of the queue manager. It follows the *`Transaction Prepared`* log record when the commit decision has been delivered to each participant.

`Purge Queue`
This logs the fact that all messages on a queue have been purged, for example, using the MQSC command CLEAR QUEUE.

`Queue Attributes`
This logs the initialization or change of the attributes of a queue.

`Create Object`
This logs the creation of a WebSphere MQ object.

ObjName	The name of the object that was created.
UserId	The user ID performing the creation.

Delete Object
This logs the deletion of a WebSphere MQ object.

ObjName	The name of the object that was deleted.

Backing up and restoring IBM WebSphere MQ queue manager data

Backing up queue managers and queue manager data.

Periodically, you can take measures to protect queue managers against possible corruption caused by hardware failures. There are three ways of protecting a queue manager:

Back up the queue manager data
If the hardware fails, a queue manager might be forced to stop. If any queue manager log data is lost due to the hardware failure, the queue manager might be unable to restart. If you back up queue manager data you might be able to recover some, or all, of the lost queue manager data.

In general, the more frequently you back up queue manager data, the less data you lose in the event of hardware failure that results in loss of integrity of the recovery log.

To back up queue manager data, the queue manager must not be running.

To back up and restore queue manager data see:

- "Backing up queue manager data" on page 1139.
- "Restoring queue manager data" on page 1139.

Use a backup queue manager
If the hardware failure is severe, a queue manager might be unrecoverable. In this situation, if the unrecoverable queue manager has a dedicated backup queue manager, the backup queue manager can be activated in place of the unrecoverable queue manager. If it was updated regularly, the backup queue manager log can contain log data that includes the last complete log from the unrecoverable queue manager.

A backup queue manager can be updated while the existing queue manager is still running.

To create and activate a backup queue manager see:

- "Creating a backup queue manager" on page 1141.
- "Starting a backup queue manager" on page 1142.

Back up the queue manager configuration only
If the hardware fails, a queue manager might be forced to stop. If both the queue manager configuration and log data is lost due to the hardware failure, the queue manager will be unable to restart or to be recovered from the log. If you back up the queue manager configuration you would be able to recreate the queue manager and all of its objects from saved definitions.

To back up queue manager configuration, the queue manager must be running.

To back up and restore the queue manager configuration see:

- "Backing up queue manager configuration" on page 1142
- "Restoring queue manager configuration" on page 1143

Backing up queue manager data

Backing up queue manager data can help you to guard against possible loss of data caused by hardware errors.

Before you begin

Ensure that the queue manager is not running. If you try to take a backup of a running queue manager, the backup might not be consistent because of updates in progress when the files were copied. If possible, stop your queue manager by running the **endmqm -w** command (a wait shutdown), only if that fails, use the **endmqm -i** command (an immediate shutdown).

About this task

To take a backup copy of a queue manager's data, complete the following tasks:

1. Search for the directories under which the queue manager places its data and its log files, by using the information in the configuration files. For more information, see "Changing IBM WebSphere MQ and queue manager configuration information" on page 1143.

   Note: You might have some difficulty in understanding the names that appear in the directory. The names are transformed to ensure that they are compatible with the platform on which you are using WebSphere MQ. For more information about name transformations, see Understanding WebSphere MQ file names.
2. Take copies of all the queue manager's data and log file directories, including all subdirectories.

   Make sure that you do not miss any files, especially the log control file, as described in "What logs look like" on page 1123, and the configuration files as described in "Initialization and configuration files" on page 803. Some of the directories might be empty, but you need them all to restore the backup at a later date.
3. Preserve the ownerships of the files. For WebSphere MQ for UNIX and Linux systems, you can do this with the **tar** command. (If you have queues larger than 2 GB, you cannot use the **tar** command. For more information, see Enabling large queues.

   Note: When you upgrade to WebSphere MQ Version 7.5 and later, ensure to take a backup of the `.ini` file and the registry entries. The queue manager information is stored in the `.ini` file and can be used to revert to a previous version of WebSphere MQ.

Restoring queue manager data

Follow these steps to restore a backup of a queue manager's data.

Before you begin

Ensure that the queue manager is not running.

About this task

To restore a backup of a queue manager's data:

1. Find the directories under which the queue manager places its data and its log files, by using the information in the configuration files.
2. Empty the directories into which you are going to place the backed-up data.
3. Copy the backed-up queue manager data and log files into the correct places.
4. Update the configuration information files.

Check the resulting directory structure to ensure that you have all the required directories.

For more information about WebSphere MQ directories and subdirectories, see Directory structure on Windows systems and Directory content on UNIX and Linux systems .

Make sure that you have a log control file as well as the log files. Also check that the WebSphere MQ and queue manager configuration files are consistent so that WebSphere MQ can look for the restored data in the correct places.

For circular logging, back up the queue manager data and log file directories at the same time so that you can restore a consistent set of queue manager data and logs.

For linear logging, back up the queue manager data and log file directories at the same time. It is possible to restore only the queue manager data files if a corresponding complete sequence of log files is available.

Note: When you upgrade to WebSphere MQ Version 7.5 and later, ensure to take a backup of the `.ini` file and the registry entries. The queue manager information is stored in the `.ini` file and can be used to revert to a previous version of WebSphere MQ.

Results

If the data was backed up and restored correctly, the queue manager will now start.

Using a backup queue manager

An existing queue manager can have a dedicated backup queue manager.

A backup queue manager is an inactive copy of the existing queue manager. If the existing queue manager becomes unrecoverable due to severe hardware failure, the backup queue manager can be brought online to replace the unrecoverable queue manager.

The existing queue manager log files must regularly be copied to the backup queue manager to ensure that the backup queue manager remains an effective method for disaster recovery. The existing queue manager does not need to be stopped for log files to be copied, however you should only copy a log file if the queue manager has finished writing to it. Because the existing queue manager log is continually updated, there is always a slight discrepancy between the existing queue manager log and the log data copied to the backup queue manager log. Regular updates to the backup queue manager minimizes the discrepancy between the two logs.

If a backup queue manager is required to be brought online it must be activated, and then started. The requirement to activate a backup queue manager before it is started is a preventive measure to protect against a backup queue manager being started accidentally. Once a backup queue manager is activated it can no longer be updated.

For information on how to create, update, and start a backup queue manager, see the following topics:
- “Creating a backup queue manager” on page 1141
- “Updating a backup queue manager” on page 1141
- “Starting a backup queue manager” on page 1142

Creating a backup queue manager

You can only use a backup queue manager when using linear logging.

To create a backup queue manager for an existing queue manager, do the following:

1. Create a backup queue manager for the existing queue manager using the control command `crtmqm`. The backup queue manager requires the following:
   - To have the same attributes as the existing queue manager, for example the queue manager name, the logging type, and the log file size.
   - To be on the same platform as the existing queue manager.
   - To be at an equal, or higher, code level than the existing queue manager.
2. Take copies of all the existing queue manager's data and log file directories, including all subdirectories, as described in "Backing up queue manager data" on page 1139.
3. Overwrite the backup queue manager's data and log file directories, including all subdirectories, with the copies taken from the existing queue manager.
4. Execute the following control command on the backup queue manager:

   ```
   strmqm -r BackupQMName
   ```

   This flags the queue manager as a backup queue manager within WebSphere MQ, and replays all the copied log extents to bring the backup queue manager in step with the existing queue manager.

Updating a backup queue manager

To ensure that a backup queue manager remains an effective method for disaster recovery it must be updated regularly.

Regular updating lessens the discrepancy between the backup queue manager log, and the current queue manager log. There is no need to stop the queue manager to be backed up.

To update a backup queue manager, do the following:

1. Issue the following Script (MQSC) command on the queue manager to be backed up:

   ```
   RESET QMGR TYPE(ADVANCELOG)
   ```

   This stops any writing to the current log, and then advances the queue manager logging to the next log extent. This ensures you back up all information logged up to the current time.
2. Obtain the (new) current active log extent number by issuing the following Script (MQSC) command on the queue manager to be backed up:

   ```
   DIS QMSTATUS CURRLOG
   ```
3. Copy the updated log extent files from the current queue manager log directory to the backup queue manager log directory - copy all the log extents since the last update, and up to (but not including) the current extent noted in step 2. Copy only log extent files, the ones beginning with "S...".
4. Issue the following control command on the backup queue manager:

   ```
   strmqm -r BackupQMName
   ```

   This replays all the copied log extents and brings the backup queue manager into step with the queue manager. When the replay finishes you receive a message that identifies all the log extents required for restart recovery, and all the log extents required for media recovery.

   Warning: If you copy a **non-contiguous** set of logs to the backup queue manager log directory, only the logs up to the point where the first missing log is found will be replayed.

Starting a backup queue manager

You can substitute a backup queue manager for an unrecoverable queue manager.

To do this, perform the following steps:

1. Execute the following control command to activate the backup queue manager:

   ```
   strmqm -a BackupQMName
   ```

   The backup queue manager is activated. Now active, the backup queue manager can no longer be updated.
2. Execute the following control command to start the backup queue manager:

   ```
   strmqm BackupQMName
   ```

   WebSphere MQ regards this as restart recovery, and utilizes the log from the backup queue manager. During the last update to the backup queue manager replay will have occurred, therefore only the active transactions from the last recorded checkpoint are rolled back.

   When an unrecoverable queue manager is substituted for a backup queue manager some of the queue manager data from the unrecoverable queue manager can be lost. The amount of lost data is dependent on how recently the backup queue manager was last updated. The more recently the last update, the less queue manager data loss.
3. Restart all channels.

Check the resulting directory structure to ensure that you have all the required directories.

See Planning file system support for more information about WebSphere MQ directories and subdirectories.

Make sure that you have a log control file as well as the log files. Also check that the WebSphere MQ and queue manager configuration files are consistent so that WebSphere MQ can look in the correct places for the restored data.

If the data was backed up and restored correctly, the queue manager will now start.

Note: Even though the queue manager data and log files are held in different directories, back up and restore the directories at the same time. If the queue manager data and log files have different ages, the queue manager is not in a valid state and will probably not start. If it does start, your data is likely to be corrupt.

Backing up queue manager configuration

Backing up queue manager configuration can help you to rebuild a queue manager from its definitions.

To take a backup copy of a queue manager's configuration:

1. Ensure that the queue manager is running.
2.
   a. On AIX, HP-UX, Linux, Solaris, or Windows: Execute the Dump MQ Configuration command (dmpmqcfg) using the default formatting option of (-f mqsc) MQSC and all attributes (-a), use standard output redirection to store the definitions into a file, for example:

      ```
      dmpmqcfg -m MYQMGR -a > /mq/backups/MYQMGR.mqsc
      ```

Restoring queue manager configuration

Follow these steps to restore a backup of a queue manager's configuration.

To restore a backup of a queue manager's configuration:

1. Ensure that the queue manager is running. Note that the queue manager may have been recreated if damage to the data and logs is unrecoverable by other means.
2. Depending on your platform, execute one of the following commands:
   a. On AIX, HP-UX, Linux, Solaris, or Windows: Execute runmqsc against the queue manager, use standard input redirection to restore the definitions from a script file generated by the Dump MQ Configuration (dmpmqcfg) command, for example:

      ```
      runmqsc MYQMGR < /mq/backups/MYQMGR.mqsc
      ```

Related reference:

dmpmqcfg
Use the **dmpmqcfg** command to dump the configuration of a WebSphere MQ queue manager.

Changing IBM WebSphere MQ and queue manager configuration information

Change the behavior of IBM WebSphere MQ or an individual queue manager to suit the needs of your installation.

You can change IBM WebSphere MQ configuration information by changing the values specified on a set of configuration attributes (or parameters) that govern IBM WebSphere MQ.

Change attribute information by editing the IBM WebSphere MQ configuration files. On IBM WebSphere MQ for Windows and Linux (x86 and x86-64 platforms), the IBM WebSphere MQ configuration files can be edited using the IBM WebSphere MQ Explorer.

On Windows systems you can also use amqmdain to change configuration information, as described in amqmdain

To find out more about configuring IBM WebSphere MQ and queue managers for your platform, see the following subtopics:

Related concepts:

"Configuring" on page 737
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

Related information:

Planning
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Administering WebSphere MQ

Changing configuration information on Windows, UNIX and Linux systems

Configuration attributes are held in configuration files, at the level of the node and of the queue manager.

On Windows, UNIX and Linux platforms, you can change IBM WebSphere MQ configuration attributes within:

- A IBM WebSphere MQ configuration file (**mqs.ini**) to effect changes for IBM WebSphere MQ on the node as a whole. There is one mqs.ini file for each node.

- A queue manager configuration file (**qm.ini**) to effect changes for specific queue managers. There is one qm.ini file for each queue manager on the node.

Client configuration options are held separately, in the client configuration file.

A configuration file (or ***stanza*** file) contains one or more stanzas, which are groups of lines in the .ini file that together have a common function or define part of a system, such as log functions, channel functions, and installable services.

Because the IBM WebSphere MQ configuration file is used to locate the data associated with queue managers, a nonexistent or incorrect configuration file can cause some or all MQSC commands to fail. Also, applications cannot connect to a queue manager that is not defined in the IBM WebSphere MQ configuration file.

Any changes you make to a configuration file usually do not take effect until the next time the queue manager is started.

On Windows and Linux (x86 and x86-64 platforms) systems, you can edit configuration information from the IBM WebSphere MQ Explorer.

On Windows systems you can also use the the **amqmdain** command to edit the configuration files.

For more information about the configuration options on Windows, UNIX and Linux systems, see the following subtopics:

Related concepts:

"Configuring" on page 737
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

"Changing IBM WebSphere MQ and queue manager configuration information" on page 1143
Change the behavior of IBM WebSphere MQ or an individual queue manager to suit the needs of your installation.

Related reference:

"Attributes for changing WebSphere MQ configuration information" on page 1151
On WebSphere MQ for Windows systems and on WebSphere MQ for Linux (x86 and x86-64 platforms) systems, modify configuration information using the WebSphere MQ Explorer. On other systems, modify the information by editing the mqs.ini configuration file.

"Changing queue manager configuration information" on page 1157
The attributes described here modify the configuration of an individual queue manager. They override any settings for WebSphere MQ.

Related information:

Planning
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Administering WebSphere MQ

Editing configuration files

Edit configuration files using commands or a standard text editor.

Before editing a configuration file, back it up so that you have a copy you can revert to if the need arises.

You can edit configuration files either:

- Automatically, using commands that change the configuration of queue managers on the node

- Manually, using a standard text editor

You can edit the default values in the WebSphere MQ configuration files after installation.

If you set an incorrect value on a configuration file attribute, the value is ignored and an operator message is issued to indicate the problem. (The effect is the same as missing out the attribute entirely.)

When you create a new queue manager:
- Back up the WebSphere MQ configuration file
- Back up the new queue manager configuration file

Comments can be included in configuration files by adding a ";" or a "#" character before the comment text. If you want to use a ";" or a "#" character without it representing a comment, you can prefix the character with a "\" character and it will be used as part of the configuration data.

When do you need to edit a configuration file?

Edit a configuration file to recover from backup, move a queue manager, change the default queue manager or to assist IBM support.

You might need to edit a configuration file if, for example:
- You lose a configuration file. (Recover from backup if you can.)
- You need to move one or more queue managers to a new directory.
- You need to change your default queue manager; this could happen if you accidentally delete the existing queue manager.
- You are advised to do so by your IBM Support Center.

Configuration file priorities

The value of an attribute is defined in multiple places. Attributes set in commands take precedence over attributes in configuration files.

The attribute values of a configuration file are set according to the following priorities:
- Parameters entered on the command line take precedence over values defined in the configuration files
- Values defined in the qm.ini files take precedence over values defined in the mqs.ini file

The IBM WebSphere MQ configuration file, mqs.ini

The IBM WebSphere MQ configuration file, mqs.ini, contains information relevant to all the queue managers on the node. It is created automatically during installation.

On IBM WebSphere MQ for UNIX and Linux products, the data directory and log directory are always `/var/mqm` and `/var/mqm/log` respectively.

On Windows systems, the location of the data directory `mqs.ini`, and the location of the log directory, are stored in the registry, as their location can vary.

In addition, on Windows systems, the installation configuration information (contained in `mqinst.ini` on IBM WebSphere MQ for UNIX and Linux systems) is in the registry, as there is no `mqinst.ini` file on Windows.

The mqs.ini file for Windows systems is given by the WorkPath specified in the `HKLM\SOFTWARE\IBM\WepSphere MQ` key. It contains:
- The names of the queue managers
- The name of the default queue manager

- The location of the files associated with each of them

The supplied LogDefaults stanza for a new IBM WebSphere MQ installation does not contain any explicit values for the attributes. The lack of an attribute means that the default for this value is used upon creation of a new queue manager. The default values are shown for the LogDefaults stanza in Figure 180 on page 1147. A value of zero for the LogBufferPages attribute means 512.

If you require a non-default value, you must explicitly specify that value in the LogDefaults stanza.

```
#***********************************************************************#
#* Module Name: mqs.ini                                                *#
#* Type       : WebSphere MQ Machine-wide Configuration File           *#
#* Function   : Define WebSphere MQ resources for an entire machine    *#
#***********************************************************************#
#* Notes      :                                                        *#
#* 1) This is the installation time default configuration              *#
#*                                                                     *#
#***********************************************************************#
AllQueueManagers:
#***********************************************************************#
#* The path to the qmgrs directory, below which queue manager data     *#
#* is stored                                                           *#
#***********************************************************************#
DefaultPrefix=/var/mqm

LogDefaults:
   LogPrimaryFiles=3
   LogSecondaryFiles=2
   LogFilePages=4096
   LogType=CIRCULAR
   LogBufferPages=0
   LogDefaultPath=/var/mqm/log

QueueManager:
   Name=saturn.queue.manager
   Prefix=/var/mqm
   Directory=saturn!queue!manager
   InstallationName=Installation1

QueueManager:
   Name=pluto.queue.manager
   Prefix=/var/mqm
   Directory=pluto!queue!manager
   InstallationName=Installation2

DefaultQueueManager:
   Name=saturn.queue.manager

ApiExitTemplate:
   Name=OurPayrollQueueAuditor
   Sequence=2
   Function=EntryPoint
   Module=/usr/ABC/auditor
   Data=123

ApiExitCommon:
   Name=MQPoliceman
   Sequence=1
   Function=EntryPoint
   Module=/usr/MQPolice/tmqp
   Data=CheckEverything
```

Figure 180. Example of a IBM WebSphere MQ configuration file for UNIX systems

Queue manager configuration files, qm.ini

A queue manager configuration file, qm.ini, contains information relevant to a specific queue manager.

There is one queue manager configuration file for each queue manager. The qm.ini file is automatically created when the queue manager with which it is associated is created.

On UNIX and Linux systems a qm.ini file is held in the root of the directory tree occupied by the queue manager. For example, the path and the name for a configuration file for a queue manager called QMNAME is:

```
/var/mqm/qmgrs/QMNAME/qm.ini
```

On Windows systems the location of the qm.ini file is given by the WorkPath specified in the `HKLM\SOFTWARE\IBM\WebSphere MQ` key. For example, the path and the name for a configuration file for a queue manager called QMNAME is:

```
C:\Program Files\IBM\WebSphere MQ\qmgrs\QMNAME\qm.ini
```

The queue manager name can be up to 48 characters in length. However, this does not guarantee that the name is valid or unique. Therefore, a directory name is generated based on the queue manager name. This process is known as *name transformation*. For a description, see Understanding WebSphere MQ file names.

Figure 181 on page 1149 shows how groups of attributes might be arranged in a queue manager configuration file in IBM WebSphere MQ for UNIX and Linux systems.

```
#* Module Name: qm.ini                                          *#
#* Type       : WebSphere MQ queue manager configuration file    *#
#  Function   : Define the configuration of a single queue manager *#
#*                                                               *#
#*****************************************************************#
#* Notes      :                                                  *#
#* 1) This file defines the configuration of the queue manager   *#
#*                                                               *#
#*****************************************************************#

ExitPath:
   ExitsDefaultPath=/var/mqm/exits
   ExitsDefaultPath64=/var/mqm/exits64

Service:
   Name=AuthorizationService
   EntryPoints=13

ServiceComponent:
   Service=AuthorizationService
   Name=MQSeries.UNIX.auth.service
   Module=opt/mqm/bin/amqzfu
   ComponentDataSize=0

Log:
   LogPrimaryFiles=3
   LogSecondaryFiles=2
   LogFilePages=4096
   LogType=CIRCULAR
   LogBufferPages=0[1]
   LogPath=/var/mqm/log/saturn!queue!manager/

AccessMode:
   SecurityGroup=wmq\wmq

XAResourceManager:
   Name=DB2 Resource Manager Bank
   SwitchFile=/usr/bin/db2swit
   XAOpenString=MQBankDB
   XACloseString=
   ThreadOfControl=THREAD

Channels: [2]
   MaxChannels=200
   MaxActiveChannels=100
   MQIBindType=STANDARD

AccessMode:
   SecurityGroup=wmq\wmq
TCP:
   KeepAlive = Yes
   SvrSndBuffSize=32768
   SvrRcvBuffSize=32768
   Connect_Timeout=0

QMErrorLog:
   ErrorLogSize=262144
   ExcludeMessage=7234
   SuppressMessage=9001,9002,9202
   SuppressInterval=30

ApiExitLocal:
   Name=ClientApplicationAPIchecker
   Sequence=3
   Function=EntryPoint
   Module=/usr/Dev/ClientAppChecker
   Data=9.20.176.20
```

Figure 181. Example queue manager configuration file for IBM WebSphere MQ for UNIX and Linux systems

Notes for Figure 181 on page 1149:

1. The value of zero for `LogBufferPages` gives a value of 512.
2. For more information on the Channel stanza, see “Initialization and configuration files” on page 803.
3. The maximum number of XAResourceManager stanzas is limited to 255. However, you should use only a small number of stanzas to avoid transaction performance degradation.

Note: WebSphere MQ in Unix uses configuration files that have a file extension of .ini, for example, qm.ini. There are some utilities in WebSphereMQ, such as **setmqm**, that will make a temporary backup copy of the files. For example, the file qm.ini will create a backup copy named qm.ini.bak. A utility will modify the qm.ini file, store the updated file, then delete the qm.ini.bak file. If the utility cannot store the qm.ini file, then it restores the contents of qm.ini from the backup file qm.ini.bak, and then deletes the qm.ini.bak file.

If there is an existing qm.ini.bak file, the utility reverts the qm.ini file with the contents of the qm.ini.bak and deletes the qm.ini.bak file. Therefore, you should not create backup copies of the *.ini files by using a file extension of .bak, because these backup files might be deleted by WebSphere MQ utilities.

See “Changing configuration information on Windows, UNIX and Linux systems” on page 1143 for information on when your changes take effect.

Installation configuration file, mqinst.ini

UNIX and Linux systems

The installation configuration file, `mqinst.ini`, contains information about all the IBM WebSphere MQ installations on a UNIX or Linux system.

The `mqinst.ini` file is in the `/etc/opt/mqm` directory on UNIX and Linux systems. It contains information about which installation, if any, is the primary installation as well as the following information for each installation:

- The installation name
- The installation description
- The installation identifier
- The installation path

This file must not be edited or referenced directly since its format is not fixed, and could change. Instead, use the following commands to create, delete, query, and modify, the values in the mqinst.ini file:

crtmqinst to create entries.

dltmqinst to delete entries.

dspmqinst to display entries.

setmqinst to set entries.

The installation identifier, for internal use only, is set automatically and must not be changed.

Windows systems

Installation configuration information is held in the following key on Windows systems:

```
HKLM\SOFTWARE\IBM\WebSphere MQ\Installation\<InstallationName>
```

This key must not be edited or referenced directly since its format is not fixed, and could change. Instead, use the following commands to query, and modify, the values in the registry:

dspmqinst to display entries.

setmqinst to set entries.

On Windows, the **crtmqinst** and **dltmqinst** commands are not available. The installation and uninstallation processes handle the creation and deletion of the required registry entries.

Attributes for changing WebSphere MQ configuration information

On WebSphere MQ for Windows systems and on WebSphere MQ for Linux (x86 and x86-64 platforms) systems, modify configuration information using the WebSphere MQ Explorer. On other systems, modify the information by editing the mqs.ini configuration file.

See the following subtopics for attributes for specific components:

Related concepts:

"Configuring" on page 737
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

"Changing IBM WebSphere MQ and queue manager configuration information" on page 1143
Change the behavior of IBM WebSphere MQ or an individual queue manager to suit the needs of your installation.

Related reference:

"Changing queue manager configuration information" on page 1157
The attributes described here modify the configuration of an individual queue manager. They override any settings for WebSphere MQ.

Related information:

Planning
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Administering WebSphere MQ

All queue managers

Use the `General` and `Extended` WebSphere MQ properties page from the WebSphere MQ Explorer, or the `AllQueueManagers` stanza in the mqs.ini file to specify the following information about all queue managers.

`DefaultPrefix=`*`directory_name`*

This attribute specifies the path to the qmgrs directory, within which the queue manager data is kept.

If you change the default prefix for the queue manager, replicate the directory structure that was created at installation time.

In particular, you must create the qmgrs structure. Stop WebSphere MQ before changing the default prefix, and restart WebSphere MQ only after you have moved the structures to the new location and changed the default prefix.

Note: Do not delete the `/var/mqm/errors` directory on UNIX and Linux systems, or the `\errors` directory on Windows systems.

As an alternative to changing the default prefix, you can use the environment variable MQSPREFIX to override the `DefaultPrefix` for the **crtmqm** command.

Because of operating system restrictions, keep the supplied path sufficiently short so that the sum of the path length and any queue manager name is a maximum of 70 characters long.

`ConvEBCDICNewline=NL_TO_LF|TABLE|ISO`

EBCDIC code pages contain a newline (NL) character that is not supported by ASCII code pages (although some ISO variants of ASCII contain an equivalent).

Use the ConvEBCDICNewline attribute to specify how WebSphere MQ is to convert the EBCDIC NL character into ASCII format.

NL_TO_LF

Convert the EBCDIC NL character (X'15') to the ASCII line feed character, LF (X'0A'), for all EBCDIC to ASCII conversions.

NL_TO_LF is the default.

TABLE

Convert the EBCDIC NL character according to the conversion tables used on your platform for all EBCDIC to ASCII conversions.

The effect of this type of conversion might vary from platform to platform and from language to language; even on the same platform, the behavior might vary if you use different CCSIDs.

ISO

Convert:

- ISO CCSIDs using the TABLE method
- All other CCSIDs using the NL_TO_CF method

Possible ISO CCSIDs are shown in Table 133.

Table 133. List of possible ISO CCSIDs

CCSID	Code Set
819	ISO8859-1
912	ISO8859-2
915	ISO8859-5
1089	ISO8859-6
813	ISO8859-7
916	ISO8859-8
920	ISO8859-9
1051	roman8

If the ASCII CCSID is not an ISO subset, ConvEBCDICNewline defaults to NL_TO_LF.

Default queue manager

Use the `General` WebSphere MQ properties page from the WebSphere MQ Explorer, or the `DefaultQueueManager` stanza in the mqs.ini file to specify the default queue manager.

Name=*default_queue_manager*

The default queue manager processes any commands for which a queue manager name is not explicitly specified. The `DefaultQueueManager` attribute is automatically updated if you create a new default queue manager. If you inadvertently create a new default queue manager and then want to revert to the original, alter the `DefaultQueueManager` attribute manually.

Exit properties

Use the `Extended` IBM WebSphere MQ properties page from the IBM WebSphere MQ Explorer, or the `ExitProperties` stanza in the mqs.ini file to specify configuration options used by queue manager exit programs.

CLWLMode=SAFE|FAST

The cluster workload (CLWL) exit allows you to specify which cluster queue in the cluster to open in response to an MQI call (for example, **MQOPEN**, **MQPUT**). The CLWL exit runs either in FAST mode or SAFE mode depending on the value you specify on the CLWLMode attribute. If you omit the CLWLMode attribute, the cluster workload exit runs in SAFE mode.

SAFE

Run the CLWL exit in a separate process from the queue manager. This is the default.

If a problem arises with the user-written CLWL exit when running in SAFE mode, the following happens:

- The CLWL server process (amqzlwa0) fails.
- The queue manager restarts the CLWL server process.
- The error is reported to you in the error log. If an MQI call is in progress, you receive notification in the form of a return code.

The integrity of the queue manager is preserved.

Note: Running the CLWL exit in a separate process can affect performance.

FAST

Run the cluster exit inline in the queue manager process.

Specifying this option improves performance by avoiding the process switching costs associated with running in SAFE mode, but does so at the expense of queue manager integrity. You should only run the CLWL exit in FAST mode if you are convinced that there are **no** problems with your CLWL exit, and you are particularly concerned about performance.

If a problem arises when the CLWL exit is running in FAST mode, the queue manager will fail and you run the risk of the integrity of the queue manager being compromised.

Log defaults for WebSphere MQ

Use the `Default log settings` WebSphere MQ properties page from the WebSphere MQ Explorer, or the `LogDefaults` stanza in the mqs.ini file to specify information about log defaults for all queue managers.

If the stanza does not exist then the MQ defaults will be used. The log attributes are used as default values when you create a queue manager, but can be overridden if you specify the log attributes on the **crtmqm** command. See **crtmqm** for details of this command.

Once a queue manager has been created, the log attributes for that queue manager are taken from the settings described in “Queue manager logs” on page 1161.

The default prefix (specified in “All queue managers” on page 1151) and log path specified for the particular queue manager (specified in “Queue manager logs” on page 1161) allow the queue manager and its log to be on different physical drives. This is the recommended method, although by default they are on the same drive.

For information about calculating log sizes, see “Calculating the size of the log” on page 1127.

Note: The limits given in the following parameter list are limits set by WebSphere MQ. Operating system limits might reduce the maximum possible log size.

LogPrimaryFiles=3|*2-254* (Windows)|*2-510* (UNIX and Linux systems)

The log files allocated when the queue manager is created.

The minimum number of primary log files you can have is 2 and the maximum is 254 on Windows, or 510 on UNIX and Linux systems. The default is 3.

The total number of primary and secondary log files must not exceed 255 on Windows, or 511 on UNIX and Linux systems, and must not be less than 3.

The value is examined when the queue manager is created or started. You can change it after the queue manager has been created. However, a change in the value is not effective until the queue manager is restarted, and the effect might not be immediate.

LogSecondaryFiles=2|*1-253* (Windows)|*1-509* (UNIX and Linux systems)

The log files allocated when the primary files are exhausted.

The minimum number of secondary log files is 1 and the maximum is 253 on Windows, or 509 on UNIX and Linux systems. The default number is 2.

The total number of primary and secondary log files must not exceed 255 on Windows, or 511 on UNIX and Linux systems, and must not be less than 3.

The value is examined when the queue manager is started. You can change this value, but changes do not become effective until the queue manager is restarted, and even then the effect might not be immediate.

LogFilePages=*number*

The log data is held in a series of files called log files. The log file size is specified in units of 4 KB pages.

The default number of log file pages is 4096, giving a log file size of 16 MB.

On UNIX and Linux systems the minimum number of log file pages is 64, and on Windows the minimum number of log file pages is 32; in both cases the maximum number is 65 535.

Note: The size of the log files specified during queue manager creation cannot be changed for a queue manager.

LogType=CIRCULAR|LINEAR

The type of log to be used. The default is CIRCULAR.

CIRCULAR

Start restart recovery using the log to roll back transactions that were in progress when the system stopped.

See "Types of logging" on page 1124 for a fuller explanation of circular logging.

LINEAR

For both restart recovery and media or forward recovery (creating lost or damaged data by replaying the contents of the log).

See "Types of logging" on page 1124 for a fuller explanation of linear logging.

If you want to change the default, you can either edit the LogType attribute, or specify linear logging using the **crtmqm** command. You cannot change the logging method after a queue manager has been created.

LogBufferPages=0|*0-4096*

The amount of memory allocated to buffer records for writing, specifying the size of the buffers in units of 4 KB pages.

The minimum number of buffer pages is 18 and the maximum is 4096. Larger buffers lead to higher throughput, especially for larger messages.

If you specify 0 (the default), the queue manager selects the size. In WebSphere MQ Version 7.0 this is 512 (2048 KB).

If you specify a number in the range 1 through 17, the queue manager defaults to 18 (72 KB). If you specify a number in the range 18 and through 4096, the queue manager uses the number specified to set the memory allocated.

LogDefaultPath=*directory_name*

The directory in which the log files for a queue manager reside. The directory resides on a local device to which the queue manager can write and, preferably, on a different drive from the message queues. Specifying a different drive gives added protection in case of system failure.

The default is:

- <DefaultPrefix>\log for WebSphere MQ for Windows where <DefaultPrefix> is the value specified on the DefaultPrefix attribute on the All Queue Managers WebSphere MQ properties page. This value is set at install time.

- /var/mqm/log for WebSphere MQ for UNIX and Linux systems

Alternatively, you can specify the name of a directory on the **crtmqm** command using the -ld flag. When a queue manager is created, a directory is also created under the queue manager directory, and this is used to hold the log files. The name of this directory is based on the queue manager name. This ensures that the log file path is unique, and also that it conforms to any limitations on directory name lengths.

If you do not specify -ld on the **crtmqm** command, the value of the LogDefaultPath attribute in the mqs.ini file is used.

The queue manager name is appended to the directory name to ensure that multiple queue managers use different log directories.

When the queue manager is created, a LogPath value is created in the log attributes in the configuration information, giving the complete directory name for the queue manager's log. This value is used to locate the log when the queue manager is started or deleted.

LogWriteIntegrity=SingleWrite|DoubleWrite| TripleWrite
The method the logger uses to reliably write log records.

TripleWrite
This is the default method.

Note, that you can select **DoubleWrite**, but if you do so, the system interprets this as **TripleWrite**.

SingleWrite
You should use **SingleWrite**, only if the file-system or device hosting the WebSphere MQ recovery log explicitly guarantees the atomicity of 4KB writes.

That is, when a write of a 4KB page fails for any reason, the only two possible states are either the before image, or the after image. No intermediate state should be possible.

Advanced Configuration and Power Interface (ACPI)

Use the ACPI WebSphere MQ properties page from the WebSphere MQ Explorer, to specify how WebSphere MQ is to behave when the system receives a suspend request.

Windows supports the Advanced Configuration and Power Interface (ACPI) standard. This enables Windows users with ACPI enabled hardware to stop and restart channels when the system enters and resumes from suspend mode.

Note that the settings specified in the ACPI WebSphere MQ properties page are applied only when the Alert Monitor is running. The Alert Monitor icon is present on the taskbar if the Alert Monitor is running.

DoDialog=Y | N
Displays the dialog at the time of a suspend request.

DenySuspend=Y | N
Denies the suspend request. This is used if DoDialog=N, or if DoDialog=Y and a dialog cannot be displayed, for example, because your notebook lid is closed.

CheckChannelsRunning=Y | N
Checks whether any channels are running. The outcome can determine the outcome of the other settings.

The following table outlines the effect of each combination of these parameters:

DoDialog	DenySuspend	CheckChannels Running	Action
N	N	N	Accept the suspend request.
N	N	Y	Accept the suspend request.
N	Y	N	Deny the suspend request.
N	Y	Y	If any channels are running deny the suspend request; if not accept the request.
Y	N	N	Display the dialog (see Note; accept the suspend request). This is the default.
Y	N	Y	If no channels are running accept the suspend request; if they are display the dialog (see Note; accept the request).
Y	Y	N	Display the dialog (Note; deny the suspend request).
Y	Y	Y	If no channels are running accept the suspend request; if they are display the dialog (Note; deny the request).

Note: In cases where the action is to display the dialog, if the dialog cannot be displayed (for example because your notebook lid is closed), the DenySuspend option is used to determine whether the suspend request is accepted or denied.

API exits

Use the IBM WebSphere MQ Explorer or the **amqmdain** command to change the entries for API exits.

Use the `Exits` IBM WebSphere MQ properties page from the IBM WebSphere MQ Explorer, or the `ApiExitTemplate` and `ApiExitCommon` stanza in the mqs.ini file to identify API exit routines for all queue managers. On Windows systems, you can also use the **amqmdain** command to change the entries for API exits. (To identify API exit routines for individual queue managers, you use the `ApiExitLocal` stanza, as described in “API exits” on page 1171.)

For a complete description of the attributes for these stanzas, see Configuring API exits.

Queue managers

There is one `QueueManager` stanza for every queue manager. Use the stanza to specify the location of the queue manager directory.

On Windows, UNIX and Linux systems, there is one `QueueManager` stanza for every queue manager. These attributes specify the queue manager name, and the name of the directory containing the files associated with that queue manager. The name of the directory is based on the queue manager name, but is transformed if the queue manager name is not a valid file name. See, Understanding WebSphere MQ file names for more information about name transformation.

`Name=`*`queue_manager_name`*
The name of the queue manager.

`Prefix=`*`prefix`*
Where the queue manager files are stored. By default, this value is the same as the value specified on the DefaultPrefix attribute of the All Queue Managers information.

`Directory=`*`name`*
The name of the subdirectory under the `<prefix>\QMGRS` directory where the queue manager files are stored. This name is based on the queue manager name, but can be transformed if there is a duplicate name or if the queue manager name is not a valid file name.

DataPath=*path*
An explicit data path provided when the queue manager was created, this overrides Prefix and Directory as the path to the queue manager data.

InstallationName=*name*
The name of the WebSphere MQ installation associated with this queue manager. Commands from this installation must be used when interacting with this queue manager. If no InstallationName value is present, the queue manager is associated with an installation of WebSphere MQ earlier than version 7.1.

Related concepts:

"Associating a queue manager with an installation" on page 748
When you create a queue manager, it is automatically associated with the installation that issued the **crtmqm** command. On UNIX, Linux, and Windows, you can change the installation associated with a queue manager using the **setmqm** command.

Security

Use the Security stanza in the qm.ini file to specify options for the Object Authority Manager (OAM).

ClusterQueueAccessControl=RQMName|Xmitq
Set this attribute to check the access control of cluster queues or fully qualified queues hosted on cluster queue managers.

RQMName
The profiles checked for access control of remotely hosted queues are named queues or named queue manager profiles.

Xmitq
The profiles checked for access control of remotely hosted queues are resolved to the SYSTEM.CLUSTER.TRANSMIT.QUEUE.

Xmitq is the default value.

GroupModel=GlobalGroups
This attribute determines whether the OAM checks global groups when determining the group membership of a user on Windows.

The default is not to check global groups.

GlobalGroups
The OAM checks global groups.

With GlobalGroups set, the authorization commands, **setmqaut**, **dspmqaut**, and **dmpmqaut** accept global groups names; see the **setmqaut -**gparameter.

Note: Setting the ClusterQueueAcessControl=RQMName and having a custom implementation of the Authorization Service at less than MQZAS_VERSION_6 results in the queue manager not starting. In this instance, either set ClusterQueueAcessControl=Xmitq or upgrade the custom Authorization Service to MQZAS_VERSION_6 or greater.

Changing queue manager configuration information

The attributes described here modify the configuration of an individual queue manager. They override any settings for WebSphere MQ.

On UNIX and Linux systems, you modify the queue manager configuration information by editing the qm.ini configuration file. When you are defining a stanza in qm.ini, you do not need to start each item on a new line. You can use either a semicolon (;) or a hash character (#) to indicate a comment.

On Windows and Linux (x86 and x86-64 platforms) systems, you can modify some configuration information by using the IBM WebSphere MQ Explorer. However, because there are significant implications to changing installable services and their components, the installable services are read-only in the IBM WebSphere MQ Explorer. You must therefore make any changes to installable services by using **regedit** on Windows, and by editing the qm.ini file on UNIX and Linux.

For more details on changing queue manager configuration information, see the following subtopics:

Related concepts:

"Configuring" on page 737
Create one or more queue managers on one or more computers, and configure them on your development, test, and production systems to process messages that contain your business data.

"Changing IBM WebSphere MQ and queue manager configuration information" on page 1143
Change the behavior of IBM WebSphere MQ or an individual queue manager to suit the needs of your installation.

Related reference:

"Attributes for changing WebSphere MQ configuration information" on page 1151
On WebSphere MQ for Windows systems and on WebSphere MQ for Linux (x86 and x86-64 platforms) systems, modify configuration information using the WebSphere MQ Explorer. On other systems, modify the information by editing the mqs.ini configuration file.

Related information:

Planning
When planning your IBM WebSphere MQ environment, you must consider the IBM WebSphere MQ architecture that you want to configure, resource requirements, the need for logging, and backup facilities. Use the links in this topic to plan the environment where IBM WebSphere MQ runs.

Administering WebSphere MQ

Access Mode

Access Mode applies to Windows servers only. The AccessMode stanza is set by the **-a [r]** option on the **crtmqm** command. Do not change the AccessMode stanza after the queue manager has been created.

Use the access group (**-a [r]**) option of the **crtmqm** command to specify a Windows security group, members of which will be granted full access to all queue manager data files. The group can either be a local or global group, depending upon the syntax used. Valid syntax for the group name is as follows:

LocalGroup

Domain name \ GlobalGroup name

GlobalGroup name@Domain name

You must define the additional access group before running the crtmqm command with the **-a [r]** option.

If you specify the group using **-ar** instead of **-a**, the local mqm group is not granted access to the queue manager data files. Use this option, if the file system hosting the queue manager data files does not support access control entries for locally defined groups.

The group is typically a global security group, which is used to provide multi-instance queue managers with access to a shared queue manager data and logs folder. Use the additional security access group to set read and write permissions on the folder or to share containing queue manager data and log files.

The additional security access group is an alternative to using the local group named mqm to set permissions on the folder containing queue manager data and logs. Unlike the local group mqm, you can make the additional security access group a local or a global group. It must be a global group to set permissions on the shared folders that contain the data and log files used by multi-instance queue managers.

The Windows operating system checks the access permissions to read and write queue manager data and log files. It checks the permissions of the user ID that is running queue manager processes. The user ID that is checked depends on whether you started the queue manager as a service or you started it interactively. If you started the queue manager as a service, the user ID checked by the Windows system is the user ID you configured with the Prepare IBM WebSphere MQ wizard. If you started the queue manager interactively, the user ID checked by the Windows system is the user ID that ran the **strmqm** command.

The user ID must be a member of the local mqm group to start the queue manager. If the user ID is a member of the additional security access group, the queue manager can read and write files that are given permissions by using the group.

Restriction: You can specify an additional security access group only on Windows operating system. If you specify an additional security access group on other operating systems, the **crtmqm** command returns an error.

Related concepts:

"Secure unshared queue manager data and log directories and files on Windows" on page 1099
This topic describes how you can secure an alternative location for queue manager data and log files, both by using the local mqm group and an alternative security group.

"Secure shared queue manager data and log directories and files on Windows" on page 1096
This topic describes how you can secure a shared location for queue manager data and log files using a global alternative security group. You can share the location between different instances of a queue manager running on different servers.

Related tasks:

"Create a multi-instance queue manager on domain workstations or servers" on page 1073
An example shows how to set up a multi-instance queue manager on Windows on a workstation or a server that is part of a Windows domain. The server does not have to be a domain controller. The setup demonstrates the concepts involved, rather than being production scale. The example is based on Windows Server 2008. The steps might differ on other versions of Windows Server.

Related information:

crtmqm

Installable services

You change installable services on Windows by using **regedit**, and on UNIX and Linux by using the Service stanza in the qm.ini file.

Note: There are significant implications to changing installable services and their components. For this reason, the installable services are read-only in the WebSphere MQ Explorer.

To change installable services on Windows systems, use **regedit**, or on UNIX and Linux systems, use the Service stanza in the qm.ini file. For each component within a service, you must also specify the name and path of the module containing the code for that component. On UNIX and Linux systems, use the ServiceComponent stanza for this.

Name=AuthorizationService|NameService
The name of the required service.

AuthorizationService
For WebSphere MQ, the Authorization Service component is known as the object authority manager, or OAM.

The AuthorizationService stanza and its associated ServiceComponent stanza are added automatically when the queue manager is created. Add other ServiceComponent stanzas manually.

NameService
No name service is provided by default. If you require a name service, you must add the NameService stanza manually.

EntryPoints=*number-of-entries*

The number of entry points defined for the service. This includes the initialization and termination entry points.

SecurityPolicy=Default|NTSIDsRequired (WebSphere MQ for Windows only)

The SecurityPolicy attribute applies only if the service specified is the default authorization service, that is, the OAM. The SecurityPolicy attribute allows you to specify the security policy for each queue manager. The possible values are:

Default

Use the default security policy to take effect. If a Windows security identifier (NT SID) is not passed to the OAM for a particular user ID, an attempt is made to obtain the appropriate SID by searching the relevant security databases.

NTSIDsRequired

Pass an NT SID to the OAM when performing security checks.

See Windows security identifiers (SIDs) for more information.

SharedBindingsUserId=*user-type*

The SharedBindingsUserId attribute applies only if the service specified is the default authorization service, that is, the OAM. The SharedBindingsUserId attribute is used with relation to shared bindings only. This value allows you to specify whether the *UserIdentifier* field in the *IdentityContext* structure, from the MQZ_AUTHENTICATE_USER function, is the effective user Id or the real user Id. For information on the MQZ_AUTHENTICATE_USER function, see MQZ_AUTHENTICATE_USER - Authenticate user. The possible values are:

Default

The value of the *UserIdentifier* field is set as the real user Id.

Real

The value of the *UserIdentifier* field is set as the real user Id.

Effective

The value of the *UserIdentifier* field is set as the effective user Id.

FastpathBindingsUserId=*user-type*

The FastpathBindingsUserId attribute applies only if the service specified is the default authorization service, that is, the OAM. The FastpathBindingsUserId attribute is used with relation to fastpath bindings only. This value allows you to specify whether the *UserIdentifier* field in the *IdentityContext* structure, from the MQZ_AUTHENTICATE_USER function, is the effective user Id or the real user Id. For information on the MQZ_AUTHENTICATE_USER function, see MQZ_AUTHENTICATE_USER - Authenticate user. The possible values are:

Default

The value of the *UserIdentifier* field is set as the real user Id.

Real

The value of the *UserIdentifier* field is set as the real user Id.

Effective

The value of the *UserIdentifier* field is set as the effective user Id.

IsolatedBindingsUserId =*user-type*

The IsolatedBindingsUserId attribute applies only if the service specified is the default authorization service, that is, the OAM. The IsolatedBindingsUserId attribute is used with relation to isolated bindings only. This value allows you to specify whether the *UserIdentifier* field in the *IdentityContext* structure, from the MQZ_AUTHENTICATE_USER function, is the effective user Id or the real user Id. For information on the MQZ_AUTHENTICATE_USER function, see MQZ_AUTHENTICATE_USER - Authenticate user. The possible values are:

Default

The value of the *UserIdentifier* field is set as the effective user Id.

Real
The value of the *UserIdentifier* field is set as the real user Id.

Effective
The value of the *UserIdentifier* field is set as the effective user Id.

For more information about installable services and components, see Installable services and components for UNIX, Linux and Windows.

For more information about security services in general, see Setting up security on Windows, UNIX and Linux systems.

Related information:

Installable services reference information

Service components:

You must specify service component information when you add a new installable service. On Windows systems use **regedit**, and on UNIX and Linux systems use the `ServiceComponent` stanza in the qm.ini file. The authorization service stanza is present by default, and the associated component, the OAM, is active.

Specify the service components as follows:

`Service=`*`service_name`*
The name of the required service. This must match the value specified on the `Name` attribute of the Service configuration information.

`Name=`*`component_name`*
The descriptive name of the service component. This must be unique and contain only characters that are valid for the names of WebSphere MQ objects (for example, queue names). This name occurs in operator messages generated by the service. We recommend that this name begins with a company trademark or similar distinguishing string.

`Module=`*`module_name`*
The name of the module to contain the code for this component. This must be a full path name.

`ComponentDataSize=`*`size`*
The size, in bytes, of the component data area passed to the component on each call. Specify zero if no component data is required.

For more information about installable services and components, see Installable services and components for UNIX, Linux and Windows.

Queue manager logs

Use the `Log` queue manager properties page from the WebSphere MQ Explorer, or the `Log` stanza in the qm.ini file, to specify information about logging on a queue manager.

By default, these settings are inherited from the settings specified for the default log settings for the queue manager (described in “Log defaults for WebSphere MQ” on page 1153). Change these settings only if you want to configure this queue manager in a different way.

For information about calculating log sizes, see “Calculating the size of the log” on page 1127.

Note: The limits given in the following parameter list are set by WebSphere MQ. Operating system limits might reduce the maximum possible log size.

`LogPrimaryFiles=3 |`*`2-254`***` (Windows )|`***`2-510`***` (UNIX and Linux systems)`**
The log files allocated when the queue manager is created.

The minimum number of primary log files you can have is 2 and the maximum is 254 on Windows, or 510 on UNIX and Linux systems. The default is 3.

The total number of primary and secondary log files must not exceed 255 on Windows, or 511 on UNIX and Linux systems, and must not be less than 3.

The value is examined when the queue manager is created or started. You can change it after the queue manager has been created. However, a change in the value is not effective until the queue manager is restarted, and the effect might not be immediate.

LogSecondaryFiles=2 |*1-253* (Windows)|*1-509* (UNIX and Linux systems)

The log files allocated when the primary files are exhausted.

The minimum number of secondary log files is 1 and the maximum is 253 on Windows, or 509 on UNIX and Linux systems. The default number is 2.

The total number of primary and secondary log files must not exceed 255 on Windows, or 511 on UNIX and Linux systems, and must not be less than 3.

The value is examined when the queue manager is started. You can change this value, but changes do not become effective until the queue manager is restarted, and even then the effect might not be immediate.

LogFilePages=*number*

The log data is held in a series of files called log files. The log file size is specified in units of 4 KB pages.

The default number of log file pages is 4096, giving a log file size of 16 MB.

On UNIX and Linux systems the minimum number of log file pages is 64, and on Windows the minimum number of log file pages is 32; in both cases the maximum number is 65 535.

Note: The size of the log files specified during queue manager creation cannot be changed for a queue manager.

LogType=CIRCULAR |LINEAR

The type of logging to be used by the queue manager. You cannot change the type of logging to be used once the queue manager has been created. Refer to the description of the LogType attribute in “Log defaults for WebSphere MQ” on page 1153 for information about creating a queue manager with the type of logging you require.

CIRCULAR

Start restart recovery using the log to roll back transactions that were in progress when the system stopped.

See “Types of logging” on page 1124 for a fuller explanation of circular logging.

LINEAR

For both restart recovery and media or forward recovery (creating lost or damaged data by replaying the contents of the log).

See “Types of logging” on page 1124 for a fuller explanation of linear logging.

LogBufferPages=0 |*0-4096*

The amount of memory allocated to buffer records for writing, specifying the size of the buffers in units of 4 KB pages.

The minimum number of buffer pages is 18 and the maximum is 4096. Larger buffers lead to higher throughput, especially for larger messages.

If you specify 0 (the default), the queue manager selects the size. In WebSphere MQ Version 7.0 this is 512 (2048 KB).

If you specify a number in the range 1 through 17, the queue manager defaults to 18 (72 KB). If you specify a number in the range 18 through 4096, the queue manager uses the number specified to set the memory allocated.

The value is examined when the queue manager is started. The value can be increased or decreased within the limits stated. However, a change in the value is not effective until the next time the queue manager is started.

LogPath=*directory_name*

The directory in which the log files for a queue manager reside. This must exist on a local device to which the queue manager can write and, preferably, on a different drive from the message queues. Specifying a different drive gives added protection in case of system failure.

The default is:

- `C:\Program Files\IBM\WebSphere MQ\log` in WebSphere MQ for Windows .
- `/var/mqm/log` in WebSphere MQ for UNIX and Linux systems.

You can specify the name of a directory on the **crtmqm** command using the -ld flag. When a queue manager is created, a directory is also created under the queue manager directory, and this is used to hold the log files. The name of this directory is based on the queue manager name. This ensures that the log file path is unique, and also that it conforms to any limitations on directory name lengths.

If you do not specify -ld on the **crtmqm** command, the value of the `LogDefaultPath` attribute is used.

In WebSphere MQ for UNIX and Linux systems, user ID mqm and group mqm must have full authorities to the log files. If you change the locations of these files, you must give these authorities yourself. This is not required if the log files are in the default locations supplied with the product.

LogWriteIntegrity=SingleWrite|DoubleWrite| TripleWrite

The method the logger uses to reliably write log records.

TripleWrite

This is the default method.

Note, that you can select **DoubleWrite**, but if you do so, the system interprets this as **TripleWrite**.

SingleWrite

You should use **SingleWrite**, only if the file-system or device hosting the WebSphere MQ recovery log explicitly guarantees the atomicity of 4KB writes.

That is, when a write of a 4KB page fails for any reason, the only two possible states are either the before image, or the after image. No intermediate state should be possible.

Restricted mode

This option applies to UNIX and Linux systems only. The `RestrictedMode` stanza is set by the -g option on the **crtmqm** command. Do not change this stanza after the queue manager has been created. If you do not use the -g option, the stanza is not created in the qm.ini file.

There are some directories under which IBM WebSphere MQ applications create files while they are connected to the queue manager within the queue manager data directory. In order for applications to create files in these directories, they are granted world write access:

- `/var/mqm/sockets/`*`QMgrName`*`/@ipcc/ssem/`*`hostname`*`/`
- `/var/mqm/sockets/`*`QMgrName`*`/@app/ssem/`*`hostname`*`/`
- `/var/mqm/sockets/`*`QMgrName`*`/zsocketapp/`*`hostname`*`/`

where *<QMGRNAME>* is the name of the queue manager, and *<hostname>* is the host name.

On some systems, it is unacceptable to grant all users write access to these directories. For example, those users who do not need access the queue manager. Restricted mode modifies the permissions of the directories that store queue manager data. The directories can then only be accessed by members of the specified application group. The permissions on the System V IPC shared memory used to communicate with the queue manager are also modified in the same way.

The application group is the name of the group with members that have permission to do the following things:

- run MQI applications
- update all IPCC resources
- change the contents of some queue manager directories

To use restricted mode for a queue manager:

- The creator of the queue manager must be in the `mqm` group and in the application group.
- The `mqm` user ID must be in the application group.
- All users who want to administer the queue manager must be in the `mqm` group and in the application group.
- all users who want to run IBM WebSphere MQ applications must be in the application group.

Any MQCONN or MQCONNX call issued by a user who is not in the application group failed with reason code MQRC_Q_MGR_NOT_AVAILABLE.

Restricted mode operates with the IBM WebSphere MQ authorization service. Therefore you must also grant users the authority to connect to IBM WebSphere MQ and access the resources they require using the IBM WebSphere MQ authorization service.

Windows UNIX Linux Further information about configuring the IBM WebSphere MQ authorization service can be found in Setting up security on Windows, UNIX and Linux systems.

Only useIBM WebSphere MQ restricted mode when the control provided by the authorization service does not provide sufficient isolation of queue manager resources.

XA resource managers

Use the `XA resource manager` queue manager properties page from the WebSphere MQ Explorer, or the `XAResourceManager` stanza in the qm.ini file, to specify the following information about the resource managers involved in global units of work coordinated by the queue manager.

Add XA resource manager configuration information manually for each instance of a resource manager participating in global units of work; no default values are supplied.

See Database coordination for more information about resource manager attributes.

Name=*name* (mandatory)

This attribute identifies the resource manager instance.

The `Name` value can be up to 31 characters in length. You can use the name of the resource manager as defined in its XA-switch structure. However, if you are using more than one instance of the same resource manager, you must construct a unique name for each instance. You can ensure uniqueness by including the name of the database in the `Name` string, for example.

WebSphere MQ uses the `Name` value in messages and in output from the **dspmqtrn** command.

Do not change the name of a resource manager instance, or delete its entry from the configuration information, once the associated queue manager has started and the resource manager name is in effect.

SwitchFile=*name* (mandatory)

The fully-qualified name of the load file containing the resource manager's XA switch structure.

If you are using a 64-bit queue manager with 32-bit applications, the `name` value should contain only the base name of the load file containing the resource manager's XA switch structure.

The 32-bit file will be loaded into the application from the path specified by **ExitsDefaultPath**.

The 64-bit file will be loaded into the queue manager from the path specified by **ExitsDefaultPath64**.

XAOpenString=*string* (optional)
The string of data to be passed to the resource manager's xa_open entry point. The contents of the string depend on the resource manager itself. For example, the string could identify the database that this instance of the resource manager is to access. For more information about defining this attribute, see:

- Adding resource manager configuration information for DB2
- Adding resource manager configuration information for Oracle
- Adding resource manager configuration information for Sybase
- Adding resource manager configuration information for Informix®

and consult your resource manager documentation for the appropriate string.

XACloseString=*string* (optional)
The string of data to be passed to the resource manager's xa_close entry point. The contents of the string depend on the resource manager itself. For more information about defining this attribute, see:

- Adding resource manager configuration information for DB2
- Adding resource manager configuration information for Oracle
- Adding resource manager configuration information for Sybase
- Adding resource manager configuration information for Informix

and consult your database documentation for the appropriate string.

ThreadOfControl=THREAD|PROCESS
This attribute is mandatory for WebSphere MQ for Windows. The queue manager uses this value for serialization when it needs to call the resource manager from one of its own multithreaded processes.

THREAD
The resource manager is fully *thread aware*. In a multithreaded WebSphere MQ process, XA function calls can be made to the external resource manager from multiple threads at the same time.

PROCESS
The resource manager is not *thread safe*. In a multithreaded WebSphere MQ process, only one XA function call at a time can be made to the resource manager.

The `ThreadOfControl` entry does not apply to XA function calls issued by the queue manager in a multithreaded application process. In general, an application that has concurrent units of work on different threads requires this mode of operation to be supported by each of the resource managers.

Attributes of channels stanzas

These attributes determine the configuration of a channel.

This information is not applicable to WebSphere MQ for the z/OS platform.

Use the `Channels` queue manager properties page from the WebSphere MQ Explorer, or the `CHANNELS` stanza in the `qm.ini` file, to specify information about channels.

MaxChannels=100 |*number*
The maximum number of *current* channels allowed.

The value must be in the range 1 - 65535. The default is 100.

MaxActiveChannels=*MaxChannels_value*
The maximum number of channels allowed to be *active* at any time. The default is the value specified for the MaxChannels attribute.

MaxInitiators=3 |*number*
The maximum number of initiators. The default and maximum value is 3.

MQIBindType=FASTPATH|STANDARD
The binding for applications:

FASTPATH
Channels connect using **MQCONNX** FASTPATH; there is no agent process.

STANDARD
Channels connect using STANDARD.

PipeLineLength=1 |*number*
The maximum number of concurrent threads a channel will use. The default is 1. Any value greater than 1 is treated as 2.

When you use pipelining, configure the queue managers at both ends of the channel to have a *PipeLineLength* greater than 1.

Note: Pipelining is only effective for TCP/IP channels.

AdoptNewMCA=NO|SVR|SDR|RCVR|CLUSRCVR|ALL|FASTPATH
If WebSphere MQ receives a request to start a channel, but finds that an instance of the channel is already running, in some cases the existing channel instance must be stopped before the new one can start. The AdoptNewMCA attribute allows you to control which types of channels can be ended in this way.

If you specify the AdoptNewMCA attribute for a particular channel type, but the new channel fails to start because a matching channel instance is already running:

1. The new channel tries to stop the previous one by requesting it to end.
2. If the previous channel server does not respond to this request by the time the AdoptNewMCATimeout wait interval expires, the thread or process for the previous channel server is ended.
3. If the previous channel server has not ended after step 2, and after the AdoptNewMCATimeout wait interval expires for a second time, WebSphere MQ ends the channel with a CHANNEL IN USE error.

The AdoptNewMCA functionality applies to server, sender, receiver, and cluster-receiver channels. In the case of a sender or server channel, only one instance of a channel with a particular name can be running in the receiving queue manager. In the case of a receiver or cluster-receiver channel, multiple instances of a channel with a particular name might be running in the receiving queue manager, but only one instance can run at any one time from a particular remote queue manager.

Note: AdoptNewMCA is not supported on requester or server-connection channels.

Specify one or more values, separated by commas or blanks, from the following list:

NO The AdoptNewMCA feature is not required. This is the default.

SVR
Adopt server channels.

SDR
Adopt sender channels.

RCVR
Adopt receiver channels.

CLUSRCVR
Adopt cluster receiver channels.

ALL
Adopt all channel types except FASTPATH channels.

FASTPATH
Adopt the channel if it is a FASTPATH channel. This happens only if the appropriate channel type is also specified, for example: `AdoptNewMCA=RCVR,SVR,FASTPATH`.

Attention!: The AdoptNewMCA attribute might behave in an unpredictable fashion with FASTPATH channels. Exercise great caution when enabling the AdoptNewMCA attribute for FASTPATH channels.

AdoptNewMCATimeout=60 |1 - 3600
The amount of time, in seconds, that the new channel instance waits for the old channel instance to end. Specify a value in the range 1 - 3600. The default value is 60.

AdoptNewMCACheck=QM|ADDRESS|NAME|ALL
The type of checking required when enabling the `AdoptNewMCA` attribute. If possible, perform full checking to protect your channels from being shut down, inadvertently or maliciously. At the very least, check that the channel names match.

Specify one or more of the following values, separated by commas or blanks in the case of *QM*, *NAME*, or *ALL*:

QM Check that the queue manager names match.

Note that the queue manager name itself is matched, not the QMID.

ADDRESS
Check the communications source IP address. For example, the TCP/IP address.

Note: Comma separated CONNAME values apply to target addresses and are, therefore, not relevant to this option.

In the case that a multi-instance queue manager fails over from `hosta` to `hostb`, any outbound channels from that queue manager will use the source IP address of `hostb`. If this is different from `hosta`, then AdoptNewMCACheck=*ADDRESS* fails to match.

You can use SSL or TLS with mutual authentication to prevent an attacker from disrupting an existing running channel. Alternatively, use an HACMP type solution with IP-takeover instead of multi-instance queue managers, or use a network load balancer to mask the source IP address.

NAME
Check that the channel names match.

ALL
Check for matching queue manager names, the communications address, and for matching channel names.

The default is `AdoptNewMCACheck=NAME,ADDRESS,QM`.

Related concepts:

"Channel states" on page 789
A channel can be in one of many states at any time. Some states also have substates. From a given state a channel can move into other states.

TCP, LU62, NETBIOS, and SPX

Use these queue manager properties pages, or stanzas in the qm.ini file, to specify network protocol configuration parameters. They override the default attributes for channels.

TCP

Use the TCP queue manager properties page from the WebSphere MQ Explorer, or the TCP stanza in the qm.ini file, to specify Transmission Control Protocol/Internet Protocol (TCP/IP) configuration parameters.

Port=1414|*port_number*

The default port number, in decimal notation, for TCP/IP sessions. The *well known* port number for WebSphere MQ is 1414.

Library1=*DLLName1* (WebSphere MQ for Windows only)

The name of the TCP/IP sockets DLL.

The default is WSOCK32.

KeepAlive=NO|YES

Switch the KeepAlive function on or off. KeepAlive=YES causes TCP/IP to check periodically that the other end of the connection is still available. If it is not, the channel is closed.

ListenerBacklog=number

Override the default number of outstanding requests for the TCP/IP listener.

When receiving on TCP/IP, a maximum number of outstanding connection requests is set. This can be considered to be a *backlog* of requests waiting on the TCP/IP port for the listener to accept the request. The default listener backlog values are shown in Table 134.

Table 134. Default outstanding connection requests (TCP)

Platform	Default ListenerBacklog value
Windows Server	100
Windows Workstation	5
Linux	100
Solaris	100
HP-UX	20
AIX V4.2 or later	100
AIX V4.1 or earlier	10

Note: Some operating systems support a larger value than the default shown. Use this to avoid reaching the connection limit.

Conversely, some operating systems might limit the size of the TCP backlog, so the effective TCP backlog could be smaller than requested here.

If the backlog reaches the values shown in Table 134, the TCP/IP connection is rejected and the channel cannot start. For message channels, this results in the channel going into a RETRY state and retrying the connection at a later time. For client connections, the client receives an MQRC_Q_MGR_NOT_AVAILABLE reason code from **MQCONN** and retries the connection at a later time.

SvrSndBuffSize=32768|number
The size in bytes of the TCP/IP send buffer used by the server end of a client-connection server-connection channel.

SvrRcvBuffSize=32768|number
The size in bytes of the TCP/IP receive buffer used by the server end of a client-connection server-connection channel.

Connect_Timeout=0|number
The number of seconds before an attempt to connect the socket times out. The default value of zero specifies that there is no connect timeout.

LU62 (WebSphere MQ for Windows only)
Use the LU6.2 queue manager properties page from the WebSphere MQ Explorer, or the LU62 stanza in the qm.ini file, to specify SNA LU 6.2 protocol configuration parameters.

TPName
The TP name to start on the remote site.

Library1=*DLLName 1*
The name of the APPC DLL.

The default value is WCPIC32.

Library2=*DLLName2*
The same as Library1, used if the code is stored in two separate libraries.

The default value is WCPIC32.

NETBIOS (WebSphere MQ for Windows only)
Use the Netbios queue manager properties page from the WebSphere MQ Explorer, or the NETBIOS stanza in the qm.ini file, to specify NetBIOS protocol configuration parameters.

LocalName=*name*
The name by which this machine is known on the LAN.

AdapterNum=0|*adapter_number*
The number of the LAN adapter. The default is adapter 0.

NumSess=1|*number_of_sessions*
The number of sessions to allocate. The default is 1.

NumCmds=1|*number_of_commands*
The number of commands to allocate. The default is 1.

NumNames=1|*number_of_names*
The number of names to allocate. The default is 1.

Library1=*DLLName1*
The name of the NetBIOS DLL.

The default value is NETAPI32.

SPX (WebSphere MQ for Windows only)
Use the SPX queue manager properties page from the WebSphere MQ Explorer, or the SPX stanza in the qm.ini file, to specify SPX protocol configuration parameters.

Socket=5E86|*socket_number*
The SPX socket number in hexadecimal notation. The default is X'5E86'.

BoardNum=0|*adapter_number*
The LAN adapter number. The default is adapter 0.

KeepAlive=NO|YES
Switch the KeepAlive function on or off.

KeepAlive=YES causes SPX to check periodically that the other end of the connection is still available. If it is not, the channel is closed.

Library1=*DLLName1*

The name of the SPX DLL.

The default is WSOCK32.DLL.

Library2=*DLLName2*

The same as LibraryName1, used if the code is stored in two separate libraries.

The default is WSOCK32.DLL.

ListenerBacklog=number

Override the default number of outstanding requests for the SPX listener.

When receiving on SPX, a maximum number of outstanding connection requests is set. This can be considered to be a *backlog* of requests waiting on the SPX socket for the listener to accept the request. The default listener backlog values are shown in Table 135.

Table 135. Default outstanding connection requests (SPX)

Platform	Default ListenerBacklog value
Windows Server	100
Windows Workstation	5

Note: Some operating systems support a larger value than the default shown. Use this to avoid reaching the connection limit.

Conversely, some operating systems might limit the size of the SPX backlog, so the effective SPX backlog could be smaller than requested here.

If the backlog reaches the values shown in Table 135, the SPX connection is rejected and the channel cannot start. For message channels, this results in the channel going into a RETRY state and retrying the connection at a later time. For client connections, the client receives an MQRC_Q_MGR_NOT_AVAILABLE reason code from **MQCONN** and should retry the connection at a later time.

Exit path

Use the `Exits` queue manager properties page from the WebSphere MQ Explorer, or the `ExitPath` stanza in the qm.ini file to specify the path for user exit programs on the queue manager system.

ExitsDefaultPath=*string*

The ExitsDefaultPath attribute specifies the location of:

- 32-bit channel exits for clients
- 32-bit channel exits and data conversion exits for servers
- Unqualified XA switch load files

ExitsDefaultPath64=*string*

The ExitsDefaultPath64 attribute specifies the location of:

- 64-bit channel exits for clients
- 64-bit channel exits and data conversion exits for servers
- Unqualified XA switch load files

API exits:

For a server, use the `Exits` queue manager properties page from the IBM WebSphere MQ Explorer, or the `ApiExitLocal` stanza in the `qm.ini` file to identify API exit routines for a queue manager. For a client modify the `ApiExitLocal` stanza in the mqclient.ini file to identify API exit routines for a queue manager.

On Windows systems, you can also use the amqmdain command to change the entries for API exits. (To identify API exit routines for all queue managers, you use the `ApiExitCommon` and `ApiExitTemplate` stanzas, as described in "API exits" on page 1156.)

Note, that for the API exit to work correctly, the message from the server must be sent to the client unconverted. After the API exit has processed the message, the message must then be converted on the client. This, therefore, requires that you have installed all conversion exits on the client.

For a complete description of the attributes for these stanzas, see Configuring API exits.

Queue manager error logs

Use the `Extended` queue manager properties page from the WebSphere MQ Explorer, or the `QMErrorLog` stanza in the qm.ini file to tailor the operation and contents of queue manager error logs.

Attention: You can use WebSphere MQ Explorer to make the changes, only if you are using a local queue manager on the Windows platform.

ErrorLogSize=*maxsize*
Specifies the size of the queue manager error log at which it is copied to the backup. *maxsize* must be in the range 32768 through 2147483648 bytes. If **ErrorLogSize** is not specified, the default value of 2097152 bytes (2 MB) is used.

ExcludeMessage=*msgIds*
Specifies messages that are not to be written to the queue manager error log. If your WebSphere MQ system is heavily used, with many channels stopping and starting, a large number of information messages are sent to the z/OS console and hardcopy log. The WebSphere MQ-IMS bridge and buffer manager might also produce a large number of information messages, so excluding messages prevents you from receiving a large number of messages if you require it. *msqIds* contain a comma-separated list of message id's from the following:

5211 - Maximum property name length exceeded.

5973 - Distributed publish/subscribe subscription inhibited

5974 - Distributed publish/subscribe publication inhibited

6254 - The system could not dynamically load shared library

7234 - Number of messages loaded

9001 - Channel program ended normally

9002 - Channel program started

9202 - Remote host not available

9208 - Error on receive from host

9209 - Connection closed

9228 - Cannot start channel responder

9489 - SVRCONN max instances limit exceeded

9490 - SVRCONN max instances per client limit exceeded

9508 - Cannot connect to queue manager

9524 - Remote queue manager unavailable

9528 - User requested closure of channel

9558 - Remote Channel is not available

9637 - Channel is lacking a certificate

9776 - Channel was blocked by user ID

9777 - Channel was blocked by NOACCESS map

9782 - Connection was blocked by address

9999 - Channel program ended abnormally

SuppressMessage=*msgIds*

Specifies messages that are written to the queue manager error log once only in a specified time interval. If your WebSphere MQ system is heavily used, with many channels stopping and starting, a large number of information messages are sent to the z/OS console and hardcopy log. The WebSphere MQ-IMS bridge and buffer manager might also produce a large number of information messages, so suppressing messages prevents you from receiving a number of repeating messages if you require it. The time interval is specified by **SuppressInterval**. *msqIds* contain a comma-separated list of message id's from the following:

5211 - Maximum property name length exceeded.

5973 - Distributed publish/subscribe subscription inhibited

5974 - Distributed publish/subscribe publication inhibited

6254 - The system could not dynamically load shared library

7234 - Number of messages loaded

9001 - Channel program ended normally

9002 - Channel program started

9202 - Remote host not available

9208 - Error on receive from host

9209 - Connection closed

9228 - Cannot start channel responder

9489 - SVRCONN max instances limit exceeded

9490 - SVRCONN max instances per client limit exceeded

9508 - Cannot connect to queue manager

9524 - Remote queue manager unavailable

9528 - User requested closure of channel

9558 - Remote Channel is not available

9637 - Channel is lacking a certificate

9776 - Channel was blocked by user ID

9777 - Channel was blocked by NOACCESS map

9782 - Connection was blocked by address

9999 - Channel program ended abnormally

If the same message id is specified in both **SuppressMessage** and **ExcludeMessage**, the message is excluded.

SuppressInterval=*length*

Specifies the time interval, in seconds, in which messages specified in **SuppressMessage** are written to the queue manager error log once only. *length* must be in the range 1 through 86400 seconds. If **SuppressInterval** is not specified, the default value of 30 seconds is used.

Queue manager default bind type

Use the Extended queue manager properties page from the WebSphere MQ Explorer, or the Connection stanza in the qm.ini file to specify the default bind type.

DefaultBindType=SHARED|ISOLATED

If DefaultBindType is set to ISOLATED, applications and the queue manager run in separate processes, and no resources are shared between them.

If DefaultBindType is set to SHARED, applications and the queue manager run in separate processes, but some resources are shared between them.

The default is SHARED.

SSL and TLS stanza of the queue manager configuration file

Use the SSL stanza of the queue manager configuration file to configure SSL or TLS channels on your queue manager.

Online Certificate Status Protocol (OCSP)

A certificate can contain an AuthorityInfoAccess extension. This extension specifies a server to be contacted through Online Certificate Status Protocol (OCSP). To allow SSL or TLS channels on your queue manager to use AuthorityInfoAccess extensions, ensure that the OCSP server named in them is available, is correctly configured, and is accessible over the network. For more information, see Working with revoked certificates.

CrlDistributionPoint (CDP)

A certificate can contain a CrlDistributionPoint extension. This extension contains a URL which identifies both the protocol used to download a certificate revocation list (CRL) and also the server to be contacted.

If you want to allow SSL or TLS channels on your queue manager to use CrlDistributionPoint extensions, ensure that the CDP server named in them is available, correctly configured, and accessible over the network.

The SSL Stanza

Use the SSL stanza in the qm.ini file to configure how SSL or TLS channels on your queue manager attempts to use the following facilities, and how they react if problems occur when using them.

In each of the following cases, if the value supplied is not one of the valid values listed, then the default value is taken. No error messages are written mentioning that an invalid value is specified.

CDPCheckExtensions=YES|NO

CDPCheckExtensions specifies whether SSL or TLS channels on this queue manager try to check CDP servers that are named in CrlDistributionPoint certificate extensions.

- YES: SSL or TLS channels try to check CDP servers to determine whether a digital certificate is revoked.
- NO: SSL or TLS channels do not try to check CDP servers. This value is the default.

OCSPAuthentication=REQUIRED|WARN|OPTIONAL

OCSPAuthentication specifies the action to be taken when a revocation status cannot be determined from an OCSP server.

If OCSP checking is enabled, an SSL or TLS channel program attempts to contact an OCSP server.

If the channel program is unable to contact any OCSP servers, or if no server can provide the revocation status of the certificate, then the value of the OCSPAuthentication parameter is used.

- `REQUIRED`: Failure to determine the revocation status causes the connection to be closed with an error. This value is the default.
- `WARN`: Failure to determine the revocation status causes a warning message to be written in the queue manager error log, but the connection is allowed to proceed.
- `OPTIONAL`: Failure to determine the revocation status allows the connection to proceed silently. No warnings or errors are given.

`OCSPCheckExtensions=YES|NO`

OCSPCheckExtensions specifies whether SSL and TLS channels on this queue manager try to check OCSP servers that are named in AuthorityInfoAccess certificate extensions.

- `YES`: SSL and TLS channels try to check OCSP servers to determine whether a digital certificate is revoked. This value is the default.
- `NO`: SSL and TLS channels do not try to check OCSP servers.

`SSLHTTPProxyName=``*string*`

The string is either the host name or network address of the HTTP Proxy server that is to be used by GSKit for OCSP checks. This address can be followed by an optional port number, enclosed in parentheses. If you do not specify the port number, the default HTTP port, 80, is used. On the HP-UX PA-RISC and Sun Solaris SPARC platforms, and for 32-bit clients on AIX, the network address can only be an IPv4 address; on other platforms it can be an IPv4 or IPv6 address.

This attribute might be necessary if, for example, a firewall prevents access to the URL of the OCSP responder.

Exit properties

Use the Cluster queue manager properties page from the IBM WebSphere MQ Explorer, or the ExitPropertiesLocal stanza in the qm.ini file, to specify information about exit properties on a queue manager. Alternatively, you can set it using the **`amqmdain`** command.

By default, this setting is inherited from the CLWLMode attribute in the ExitProperties stanza of the machine-wide configuration (described in “Exit properties” on page 1152). Change this setting only if you want to configure this queue manager in a different way. This value can be overridden for individual queue managers using the cluster workload mode attribute on the Cluster queue manager properties page.

`CLWLMode=SAFE|FAST`

The cluster workload (CLWL) exit allows you to specify which cluster queue in the cluster to open in response to an MQI call (for example, **MQOPEN**, **MQPUT**). The CLWL exit runs either in FAST mode or SAFE mode depending on the value you specify on the CLWLMode attribute. If you omit the CLWLMode attribute, the cluster workload exit runs in SAFE mode.

`SAFE`

Run the CLWL exit in a separate process from the queue manager. This is the default.

If a problem arises with the user-written CLWL exit when running in SAFE mode, the following happens:

- The CLWL server process (amqzlwa0) fails.
- The queue manager restarts the CLWL server process.
- The error is reported to you in the error log. If an MQI call is in progress, you receive notification in the form of a return code.

The integrity of the queue manager is preserved.

Note: Running the CLWL exit in a separate process can affect performance.

`FAST`

Run the cluster exit inline in the queue manager process.

Specifying this option improves performance by avoiding the process switching costs associated with running in SAFE mode, but does so at the expense of queue manager integrity. You should only run the CLWL exit in FAST mode if you are convinced that there are **no** problems with your CLWL exit, and you are particularly concerned about performance.

If a problem arises when the CLWL exit is running in FAST mode, the queue manager will fail and you run the risk of the integrity of the queue manager being compromised.

Subpool

This stanza is created by WebSphere MQ. Do not change it.

The stanza Subpool, and attribute ShortSubpoolName within that stanza, are written automatically by WebSphere MQ when you create a queue manager. WebSphere MQ chooses a value for ShortSubpoolName. Do not alter this value.

The name corresponds to a directory and symbolic link created inside the /var/mqm/sockets directory, which WebSphere MQ uses for internal communications between its running processes.

Configuring HP Integrity NonStop Server

Use this information to help you to configure your IBM WebSphere MQ client for HP Integrity NonStop Server installation.

For details about configuring a client by using a configuration file, see "Configuring a client using a configuration file" on page 860.

For details about configuring a client by using environment variables, see "Using WebSphere MQ environment variables" on page 870.

If you are performing IBM WebSphere MQ client for HP Integrity NonStop Server operations under TMF/Gateway, see the subtopics for information about how to configure the TMF/Gateway. Included are an overview of the Gateway process, configuring the Gateway to run under Pathway, and configuring the client initialization file to enable your IBM WebSphere MQ client for HP Integrity NonStop Server to reach the TMF Gateway.

This section also contains IBM WebSphere MQ client for HP Integrity NonStop Server specific information about granting permissions to channels.

Gateway process overview

The HP NonStop Transaction Management Facility (TMF) provides services to enable a gateway process to register as a resource manager. The IBM WebSphere MQ provided TMF/Gateway process runs under Pathway.

IBM WebSphere MQ registers a single gateway process for each queue manager that is coordinated by TMF, therefore you must configure a separate TMF/Gateway for each queue manager that is to participate in TMF coordinated units of work. This registration is so that each queue manager is an independent resource manager, and for administrative purposes, registering each queue manager once with HP NonStop TMF results in an easy to understand mapping.

For multiple installations of IBM WebSphere MQ, you must nominate a single gateway process from one of these installations for each queue manager to be coordinated by TMF.

The interface to the gateway process supports any client at the same version or earlier.

For more information about administering the gateway process, see Administering HP Integrity NonStop Server.

Configuring Gateway to run under Pathway

TMF/Gateway is the interface between the HP NonStop Transaction Management Facility (TMF) and IBM WebSphere MQ that enables TMF to be the transaction coordinator for IBM WebSphere MQ transactions.

The IBM WebSphere MQ provided TMF/Gateway converts transactions from TMF coordination into eXtended Architecture (XA) transaction coordination to communicate with the remote queue manager.

You must have one TMF/Gateway per queue manager that requires coordination, and client configuration is required so that the client can connect to the correct Gateway.

The TMF/Gateway can use all the mechanisms available to the client to communicate with a queue manager. Configure the TMF/Gateway in the way you would for your other applications.

The TMF/Gateway is not a HP Integrity NonStop Server process pair and is designed to run in a Pathway environment. The TMF/Gateway creates permanent resources within TMF, which it reuses on subsequent runs, therefore the TMF/Gateway must always be run under the same user authority.

Defining the serverclass

TMF/Gateway is hosted as a serverclass within a Pathway environment. To define the serverclass, you must set the following server attributes:

PROCESSTYPE=OSS
Specifies the type of servers in the serverclass. The Gateway process is a multi-threaded OSS program. This attribute is mandatory, and must be set to OSS.

MAXSERVERS=1
Specifies the maximum number of server processes in this serverclass that can run at the same time. There can be only a single Gateway process for any queue manager. This attribute is mandatory and must be set to 1.

NUMSTATIC=1
Specifies the maximum number of static servers within this serverclass. The Gateway process must be run as a static server. This attribute is mandatory and must be set to 1.

TMF=ON
Specifies whether servers in this serverclass can lock and update data files that are audited by the TMF subsystem. The Gateway process participates in the TMF transactions of IBM WebSphere MQ client applications therefore this attribute must be set to ON.

PROGRAM=<OSS installation path>/opt/mqm/bin/runmqtmf
For IBM WebSphere MQ client for IBM WebSphere MQ, this attribute must be runmqtmf. This attribute must be the absolute OSS path name. Case is significant.

ARGLIST=-*m<QMgr name>*[,-c<channel name>][,-p<port>][,-h<host name>][,-n<max threads>]
These attributes provide parameters to the Gateway process, where:

- QMgrName is the name of the queue manager for this Gateway process. If you are using a queue sharing group (or other port distribution technology), this parameter must be targeted to a specific queue manager. This parameter is mandatory.
- channel name is the name of the server channel on the queue manager to be used by the Gateway process. This parameter is optional.
- port is the TCP/IP port for the queue manager. This parameter is optional.
- host name is the host name for the queue manager. This parameter is optional.
- max threads is the maximum number of worker threads that are created by the Gateway process. This parameter can be a value of 10 or greater. The lowest value that is used is 10 even if a value lower than 10 is specified. If no value is provided, the Gateway process creates up to a maximum of 50 threads.

Use the -c, -p, and -h attributes as an alternative method of providing connection information to the Gateway, in addition to that described in “Configuring the TMF/Gateway using environment variables.” If you specify one or more, but not all of the -c, -p, and -h attributes, then those attributes that you do not specify default to the following values:

- channel name defaults to SYSTEM.DEF.SVRCONN
- host name defaults to localhost
- port defaults to 1414

If any of the parameters you supply are invalid, the TMF/Gateway issues diagnostic message AMQ5379 to the error log and terminates.

OWNER=ID
The user ID under which the Gateway runs and that must be granted connect authority to the queue manager.

SECURITY="value"
Specifies the users, in relation to the Owner attribute, who can access the Gateway from a IBM WebSphere MQ client application.

LINKDEPTH and **MAXLINKS** must be configured with values appropriate for the expected number of IBM WebSphere MQ client applications that might want to concurrently communicate with the Gateway. If these values are set too low, you might see occurrences of the error message AMQ5399 issued from client applications.

For more information about these server attributes, see the *HP NonStop TS/MP 2.5 System Management Manual*.

Configuring the TMF/Gateway using environment variables

One of the most commonly used methods to define the TMF/Gateway is to set the MQSERVER environment variable, for example:

```
ENV MQSERVER=<channel name>/<transport>/<host name>(<listener port>)
```

ENV at the beginning of the command is Pathway notation.

Configuring the client initialization file

If you are using the HP NonStop Transaction Management Facility (TMF), you must have an IBM WebSphere MQ client initialization file to enable your IBM WebSphere MQ client for the HP Integrity NonStop Server to reach the TMF Gateway.

An IBM WebSphere MQ client initialization file for HP Integrity NonStop Server can be held in a number of locations, for more information, see “Location of the client configuration file” on page 861.

For details of the contents of the configuration file, together with an example, see “Configuring a client using a configuration file” on page 860. Use the TMF stanza to specify the TMF queue manager and server details, for more information, see “TMF and TMF/Gateway stanzas” on page 869.

An example of the entries for a IBM WebSphere MQ client for HP Integrity NonStop Server is:

```
TMF:
  PathMon=$PSD1P

TmfGateway:
  QManager=MQ5B
  Server=MQ-MQ5B
```

```
TmfGateway:
  QManager=MQ5C
  Server=MQ-MQ5C
```

For more information about configuring a client using environment variables, see "Using WebSphere MQ environment variables" on page 870.

Granting permissions to channels

Granting permissions to channels on IBM WebSphere MQ client for HP Integrity NonStop Server is identical to other operating systems, however you must know the identification of the owner that the gateway is running under.

You can then use the identification of the owner of the gateway to grant appropriate permissions. The important difference is that granting permissions to queue manager channels is not under the authority of any application.

Use the ../com.ibm.mq.ref.adm.doc/q083500_.dita command to both to grant an authorization, that is, give a IBM WebSphere MQ principal or user group permission to perform an operation, and to revoke an authorization, that is, remove the permission to perform an operation.

Index

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software for use with this program.

This book contains information on intended programming interfaces that allow the customer to write programs to obtain the services of WebSphere MQ.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Important: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks

IBM, the IBM logo, ibm.com®, are trademarks of IBM Corporation, registered in many jurisdictions worldwide. A current list of IBM trademarks is available on the Web at "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Other product and service names might be trademarks of IBM or other companies.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

This product includes software developed by the Eclipse Project (http://www.eclipse.org/).

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Sending your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:

- Send an email to ibmkc@us.ibm.com
- Use the form on the web here: www.ibm.com/software/data/rcf/

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Include the following information:

- Your name and address
- Your email address
- Your telephone or fax number
- The publication title and order number
- The topic and page number related to your comment
- The text of your comment

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

Thank you for your participation.

IBM®