This is the first of two modules discussing administration of publish/subscribe in WebSphere MQ V7.

Agenda

- Introduction
- Topic tree administration
- Subscription administration
- Additional administrative topics
- Summary

This module introduces the concept of publish/subscribe. It explains topic administration. Subscription administration and administrative topics is covered in Publish/subscribe administration part 2.

Section

Introduction

This section introduces the concept of publish/subscribe and its implementation in WebSphere MQ V7.

What is publish/subscribe?

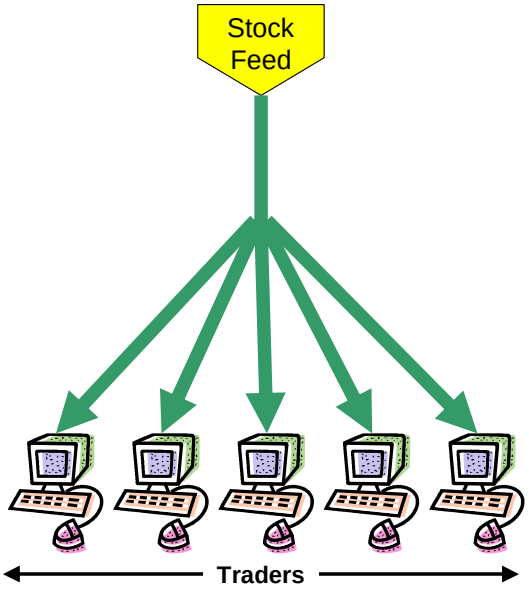
Publish/subscribe is a term used to define an application model in which the provider of some information is decoupled from the consumers of that information.

- Providers of information need have no knowledge of consumers
- Consumers of information need have no knowledge of providers
- Providers of information are called **publishers**
- Consumers of information are called **subscribers**
- New providers/consumers can be added without disruption

Publish/subscribe systems have become very popular in recent years as a way of distributing data messages from publishing computers to subscribing computers. Such systems are especially useful where data supplied by a publisher is constantly changing and a large number of subscribers needs to be quickly updated with the latest data. Perhaps the best example of where this is useful is in the distribution of stock market data.

In such systems, publisher applications of data messages do not need to know the identity or location of the subscriber applications which will receive the messages. Similarly, the subscribing applications do not need to know the identity or location of the publishing application providing their information. In this sense the providers and consumers are said to be loosely-coupled.

The classic example



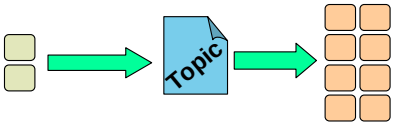
- A "feed" provides a continuous flow of information which is pushed to interested parties
- Traders consume this information and use it as a basis for the buying and selling stock

An often quoted example of a Publish/Subscribe system is one which provides stock-market information. Here a "feed" provides (publishes) a continuous flow of information containing the latest stock prices. The latest stock prices are required by traders who need this information in order to conduct trades. Traders register their interest in (subscribe to) particular stock prices and receive updates as prices change. Traders can be added and removed without disruption to the providers of the information that has no knowledge of who is receiving their information.

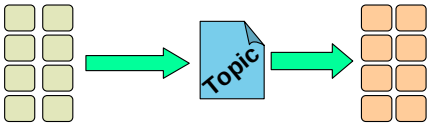
The terms "push" and "pull" describe the flow of information between applications. In this example, traders receive new information from the stock-feed as soon as a stock price changes. The information is ‘pushed’ directly to them. This pushing of information from provider to consumer differentiates a publish/subscribe from more conventional systems. This stock market example can be designed so that updated stock prices only flowed to the traders when they requested, or pulled, them from a central repository (server) of all stock prices. In such a system, the emphasis would be on the traders who request a refresh of their stock prices.

WebSphere MQ V7 supports both modes of operation.

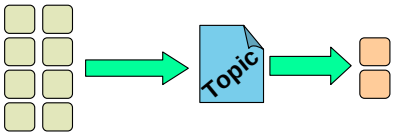
Loose-coupling with publish/subscribe



Few-to-many: Research, news tickers



Many-to-many: Prices and Quotes



Many-to-few: Orders

In the WebSphere MQ V7 publish/subscribe model the only thing which connects publishing and subscribing applications is the topic or subject which the publisher associates with his information. Publishers and subscribers need only agree on the topic to become connected to one another. Each different piece of information has its own topic associated with it. Subscribers nominate which types of information they want to receive by subscribing to specific topics.

Publishers of information are unaware of subscribers to the extent that they can publish information even if there are no subscribing applications requiring it. Publishing and subscribing are completely dynamic processes. New subscribers and new publishers can be added to the system without disruption.

With respect to a given topic, or piece of information, all possible combinations of publishers/subscribers are possible. Information about each topic can be provided by a single or multiple publishing applications. The information can be received and processed by one or more subscribing applications.

Publications and subscriptions

- Subscribers make **subscriptions** with the queue manager to register their interest in information relating to specific topics.
  - Subscribers use the MQSUB verb
- Publishers provide information about specific topics by sending **publications** to the queue manager
  - Publishers use the MQPUT verb
- The queue manager forwards each publication it receives to all subscribers with a subscription which matches the associated topic

Applications which provide information are called publishers. Applications which consume information are called subscribers.

A subscriber specifies the topic it is interested in receiving information about by specifying it on the MQSUB verb. A subscriber can make multiple subscriptions to the queue manager.

A publisher publishes its information by putting a message to a topic.

It is the job of the queue manager, or queue manager network if multiple queue managers have been connected together, to ensure that all subscribing applications with matching subscriptions to the topic being published receive the publisher's message, known as a publication.

There is a separate module that covers publish/subscribe in a multiple queue manager scenario.

Types of publications

- Events
  - Continuing succession of logically independent messages, for example:
    - trades
    - customer buying an airline ticket
  - Subscribers receive as available
- State
  - Information that is being regularly updated or replaced, for example:
    - stock prices
    - furnace temperatures
  - Queue managers can retain copy of last publication
  - Subscribers can receive immediately or check at their own initiative

As a publish/subscribe system is designed it is important to decide whether the information being published on each topic is state or is event related.

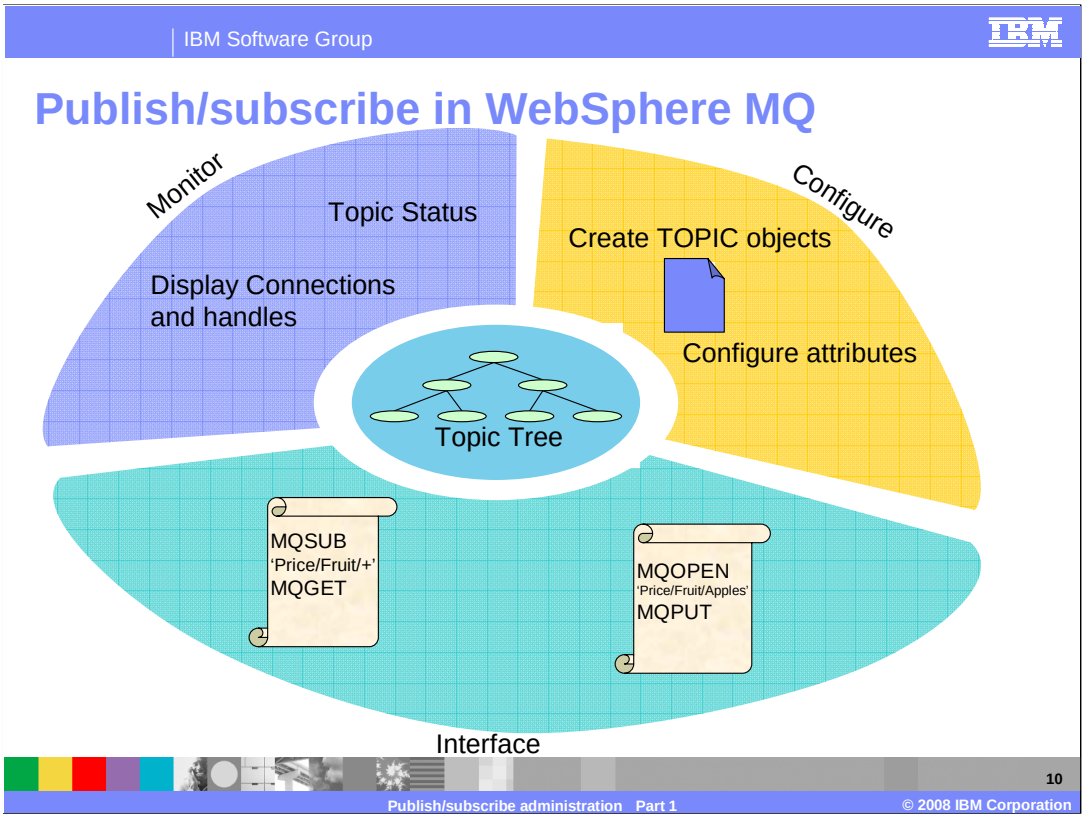
Event publications are normally independent from one another. They usually indicate that some further action or processing is needed. A subscriber missing an event can be disastrous. Generally subscriptions to event publications all need to be in place before any events are published. There can be more than one publisher of event publications for a given topic. Examples of this type of information are a stock trade or a customer buying an airline ticket.

State publications usually contain information that is updated at regular intervals. If a subscriber misses a state publication it usually is not a problem since an updated view of the state is published again. The queue manager can also be instructed to retain the last copy of a state publication. This can be sent to new subscribers to that state topic instead of when the information is published again. Typically there is only a single publisher per state topic. Examples of this type of information are a stock price or the temperature of a furnace.

Section

Topic tree administration

This section introduces the concept of topic tree and how it is administered in WebSphere MQ V7.



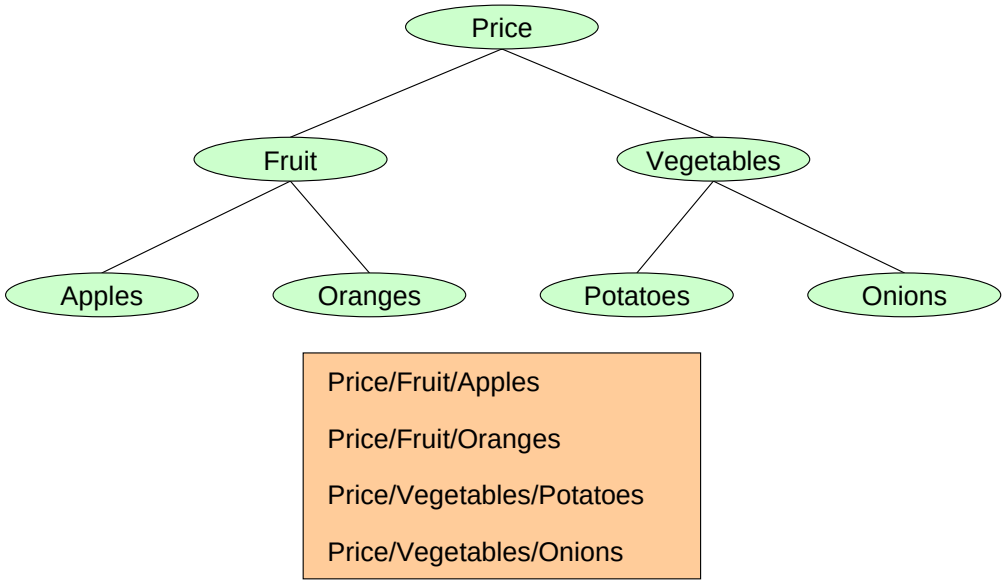
The queue manager holds a view of all the topic strings used in a hierarchical construct known as the topic tree. This topic tree is the central control point for all publish/subscribe. There are several ways to interact with the topic tree.

Administrators configure the behavior of the topic tree by defining topic objects and changing attributes on them.

You can programmatically interface with the topic tree as a subscriber using MQSUB and as a publisher using MQOPEN and MQPUT.

You can monitor the use of your topic tree by using the topic status command and the commands to display connections and their handles.

Topic strings and topic tree



Topic strings can be any characters you choose. You can, and should, add structure to you topic strings using the '/' character. This produces a topic tree with a hierarchical structure, as shown here. This hierarchical topic tree was created by the use of the topic strings shown; however, it is generally pictured as a tree.

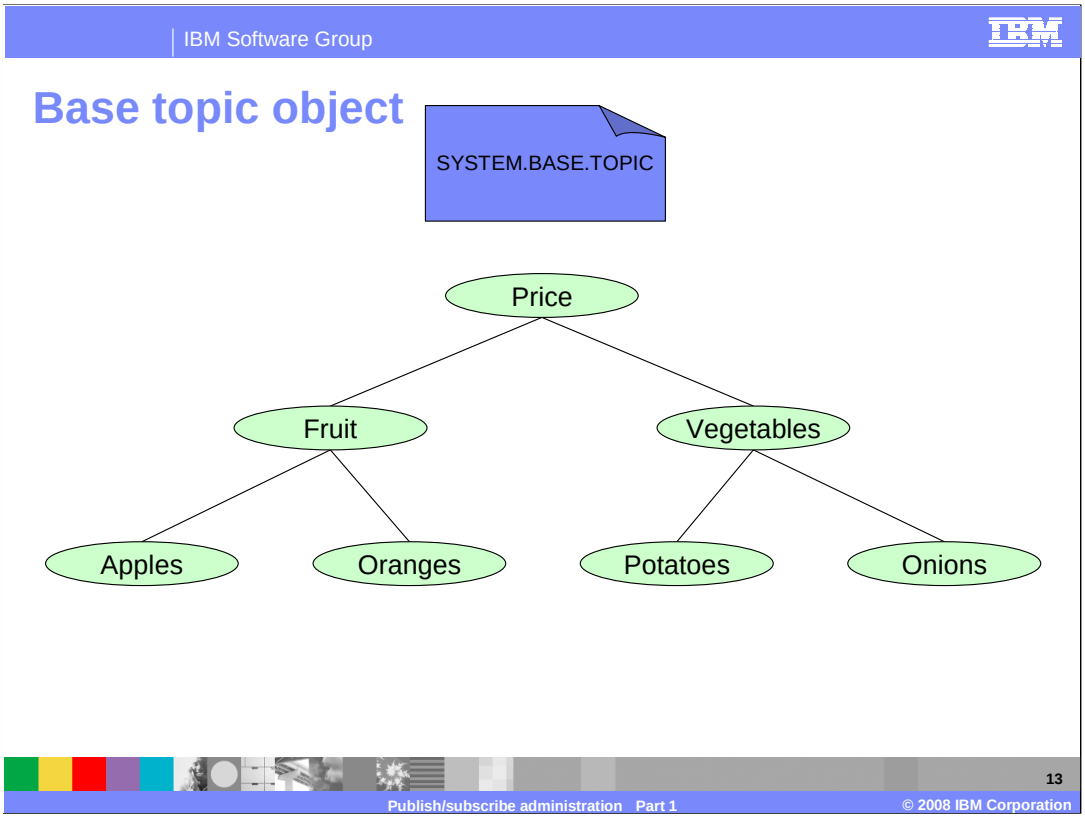
There are some special characters, apart from the '/' character, that you should avoid in your topic strings. These are '#', '+', '*' and '?'.

Topic objects

- Not necessary for publish/subscribe
- Provide an administrative control point for your topic tree
  - Configuration attributes
  - Security profiles
  - Topic tree isolation

Topic objects are a new construct in WebSphere MQ V7. They can be used to control the behavior of your topic tree.

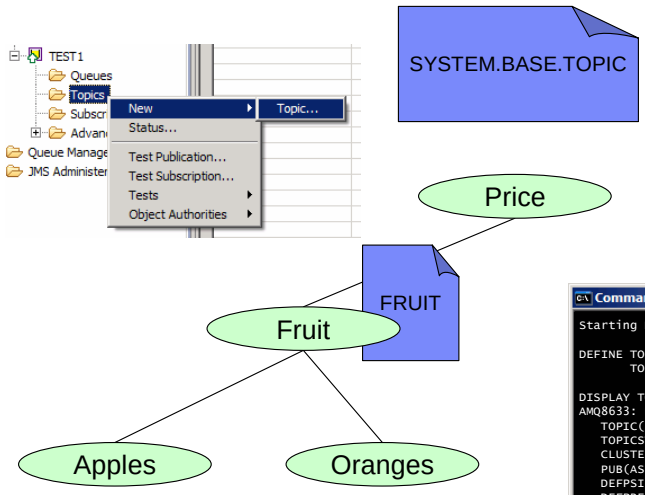
You do not need to define any topic objects in order to use publish/subscribe with WebSphere MQ V7. However, topic objects provide administrative control of your topic tree. The topic object allows configuration of the topic tree to use non-default attributes. It allows different security profile assignments to parts of your topic tree. It allows you to isolate you applications from administrative changes to the topic tree; this is similar to using remote queue and alias queue definitions.



By default there exists a base topic object, the SYSTEM.BASE.TOPIC, that contains all the settings for topic object behavior. If you define no other topic objects in your system, the behavior will be taken from this topic object. If you want your whole topic tree to behave in the same way and have no need for any other topics, you can alter this object to have the behavior you require.

If you delete this object, the queue manager acts as if the SYSTEM.BASE.TOPIC was defined with the default attributes. If you need to change that behavior, you will first need to re-define this object again.

Defining a topic object



DEFINE TOPIC

ALTER TOPIC

DELETE TOPIC

DISPLAY TOPIC

```
Command Prompt - runmqsc TEST1
Starting MQSC for queue manager TEST1.

DEFINE TOPIC(FRUIT)
       TOPICSTR( Price/Fruit ) DURSUB(NO)

DISPLAY TOPIC(FRUIT)
AMQ8633: Display topic details.
   TOPIC(FRUIT)                            TYPE(LOCAL)
   TOPICSTR(Price/Fruit)                   DESCR( )
   CLUSTER( )                              DURSUB(NO)
   PUB(ASPARENT)                           SUB(ASPARENT)
   DEFPSIST(ASPARENT)                      DEFPRTY(ASPARENT)
   DEFPRESP(ASPARENT)                      ALTDATE(2008 02 26)
   ALTTIME(15.05.22)                       PMSGDLV(ASPARENT)
   NPMSGDLV(ASPARENT)                      PUBSCOPE(ASPARENT)
   SUBSCOPE(ASPARENT)                      PROXYSUB(FIRSTUSE)
   WILDCARD(PASSTHRU)                      MDURMDL( )
   MNDURMDL( )
```


You can define a topic using MQ Explorer or using MQSC commands. This new object type has DEFINE, ALTER, DELETE and DISPLAY commands. The TOPICSTR parameter of a TOPIC object cannot be altered. Think of this attribute as the other name of the TOPIC object. You cannot alter the name of an object; you must delete and redefine it.

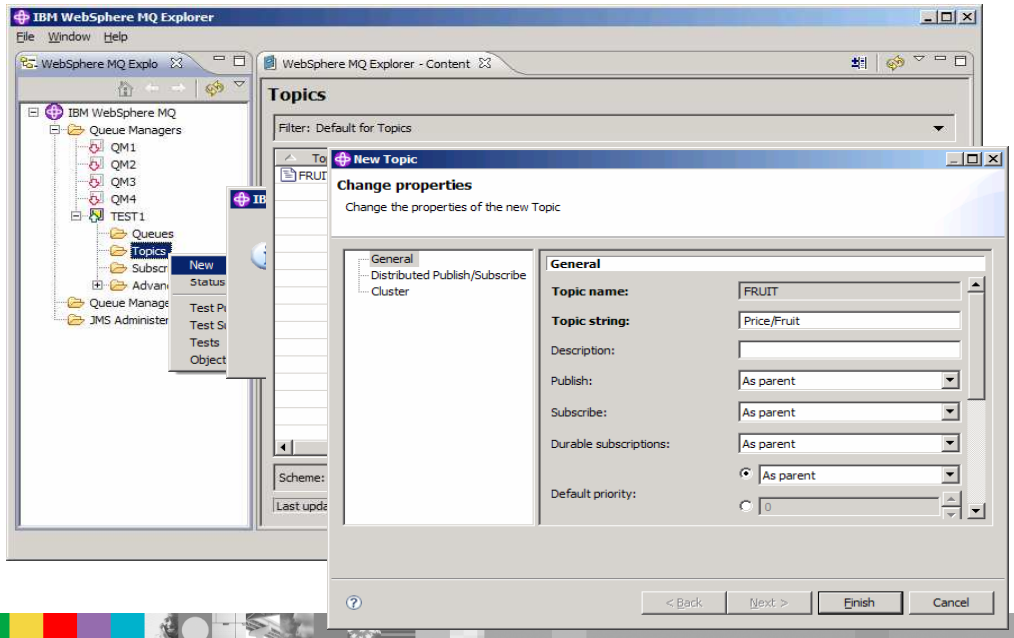
In the example here, topic FRUIT is defined in topic string Price/Fruit as a non-durable subscription. This behavior is inherited by the nodes in the topic tree below that point without the need for any further TOPIC object definitions.

Using MQSC commands

```
DEFINE TOPIC(FRUIT)
       TOPICSTR( Price/Fruit ) DURSUB(NO)

DISPLAY TOPIC(FRUIT)
AMQ8633: Display topic details.
   TOPIC(FRUIT)                            TYPE(LOCAL)
   TOPICSTR(Price/Fruit)                   DESCR( )
   CLUSTER( )                              DURSUB(NO)
   PUB(ASPARENT)                           SUB(ASPARENT)
   DEFPSIST(ASPARENT)                      DEFPRTY(ASPARENT)
   DEFPRESP(ASPARENT)                      ALTDATE(2008 02 26)
   ALTTIME(15.05.22)                       PMSGDLV(ASPARENT)
   NPMSGDLV(ASPARENT)                      PUBSCOPE(ASPARENT)
   SUBSCOPE(ASPARENT)                      PROXYSUB(FIRSTUSE)
   WILDCARD(PASSTHRU)                      MDURMDL( )
   MNDURMDL( )
```


Here is an example of defining topic FRUIT using MQSC commands. In the DISPLAY output from the object FRUIT, many of the attributes not specified in the define have the value ASPARENT and some character strings have blanks, which is the same as ASPARENT. ASPARENT means that the value for this attribute is taken from the next TOPIC object up the topic tree. If the next TOPIC up also says ASPARENT for the value being resolved, move up the tree. Eventually the top of the tree, SYSTEM.BASE.TOPIC is reached and those values are used.

Using MQ Explorer



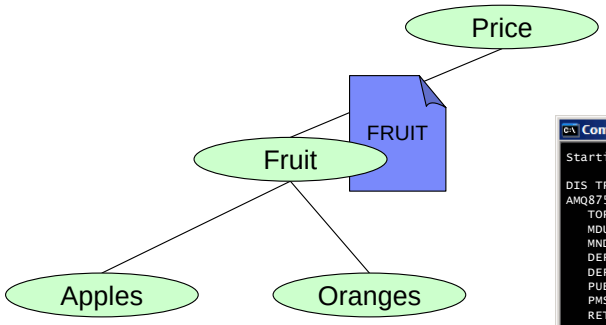
Here is an example of defining topic FRUIT in Price/Fruit with inhibited durable subscriptions using MQ Explorer.

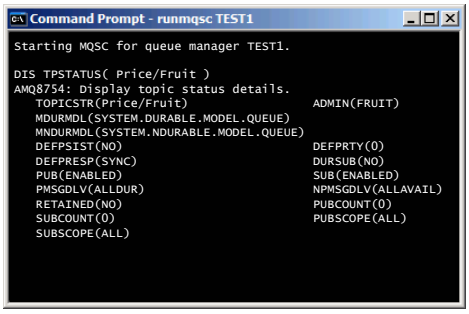
Resolving ASPARENT

Topic status:

Topic string	Publish	Subscribe	Admin topic name	Durable subscriptions
/				
Price	Allowed	Allowed		Allowed
Fruit	Allowed	Allowed	FRUIT	Inhibited
Apples	Allowed	Allowed		Inhibited
Oranges	Allowed	Allowed		Inhibited

DISPLAY TPSTATUS





The DISPLAY TPSTATUS command provides the actual values assigned to the attributes that showed ASPARENT, or blanks, in the DISPLAY TOPIC command.

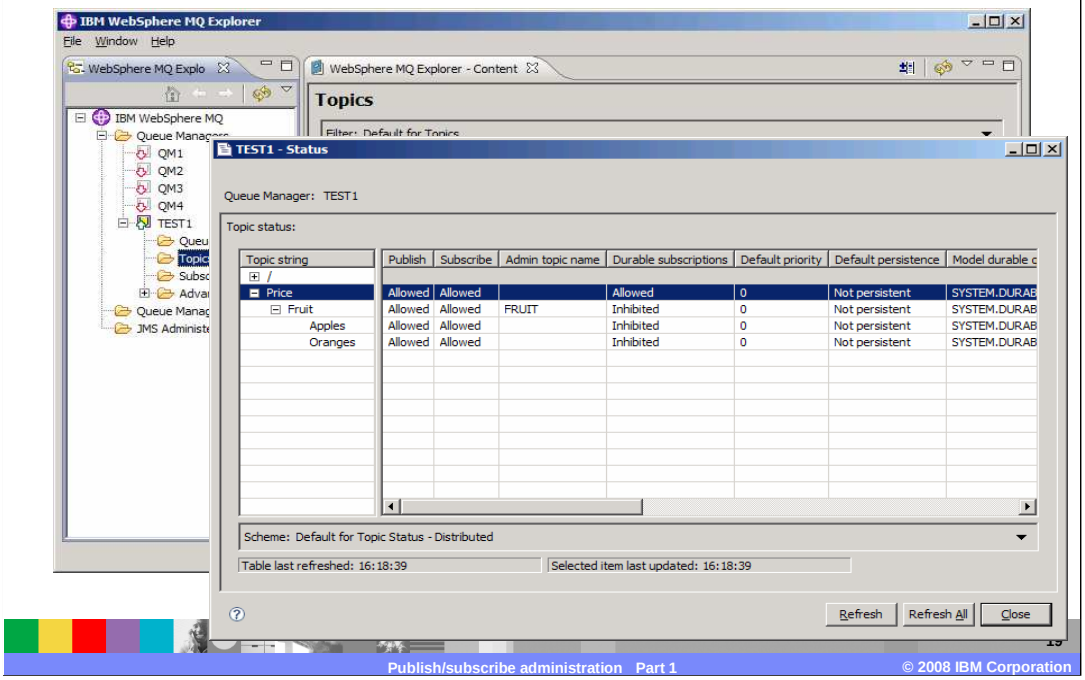
This command takes a topic string, not a topic object as its input. This means you can find the actual values that are going to be used at any point in the topic tree, not just at those points which have defined TOPIC objects.

DISPLAY TPSTATUS in MQSC

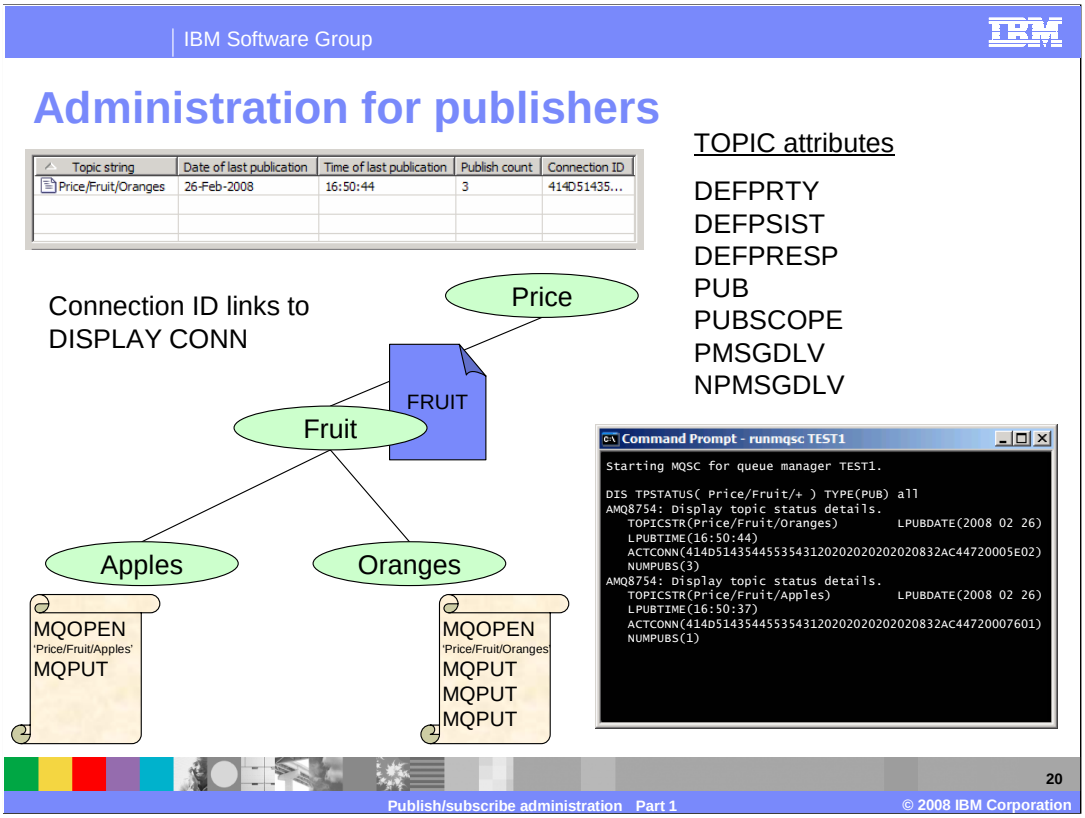
```
DIS TPSTATUS( Price/Fruit )
AMQ8754: Display topic status details.
   TOPICSTR(Price/Fruit)                   ADMIN(FRUIT)
   MDURMDL(SYSTEM.DURABLE.MODEL.QUEUE)
   MNDURMDL(SYSTEM.NDURABLE.MODEL.QUEUE)
   DEFPSIST(NO)                            DEFPRTY(0)
   DEFPRESP(SYNC)                          DURSUB(NO)
   PUB(ENABLED)                            SUB(ENABLED)
   PMSGDLV(ALLDUR)                         NPMSGDLV(ALLAVAIL)
   RETAINED(NO)                            PUBCOUNT(0)
   SUBCOUNT(0)                             PUBSCOPE(ALL)
   SUBSCOPE(ALL)
```


Here is the output of a DISPLAY TPSTATUS using MQSC.

Topic status in MQ Explorer



Here is an example of the topic status using MQ Explorer.



There are a few attributes on the topic object that are relevant to publishers. These can be defined and displayed from command line or in MQ Explorer.

MQPUT options resolution

- Topic options related to MQPUT
  - DEFPRTY (default message priority)
    - Constant DEFPRTY
  - DFTMSGPST (default message persistence)
    - Constant DEFPSIST
  - DFTPUTRESP (default put response)
    - Constant DEFPRESP
- MQPUT constants
  - MQPRI_PRIORITY_AS_TOPIC_DEF
  - MQPER_PERSISTENCE_AS_TOPIC_DEF
  - MQPMO_RESPONSE_AS_TOPIC_DEF
  - Values same as equivalent AS_Q_DEF constants

Various options on MQPUT can be resolved from the object that was opened. These are priority, persistence and asynchronous put response. The constants listed here have the same numeric value as the equivalent AS_Q_DEF constants. Using these MQPUT constants means that that the actual value is resolved from the topic object.

Topic options for publishing

- PUB
  - Controls whether messages can be published to this topic
- PUBSCOPE
  - Determines propagating of publications
    - part of a hierarchy
    - part of a publish/subscribe cluster
- PMSGDLV
  - Provides the delivery mechanism for persistent messages
- NPMSGDLV
  - Provides the delivery mechanism for non-persistent messages

The TOPIC attribute PUB determines whether publishing is allowed at this point in the topic tree. If set to DISABLED, an MQPUT call will fail with MQRC_PUT_INHIBITED. PUBSCOPE determines whether this queue manager propagates publications to queue managers as part of a hierarchy or as part of a publish/subscribe cluster. PMSGDLV and NPMSGDLV provide the delivery mechanism for persistent and non-persistent messages.

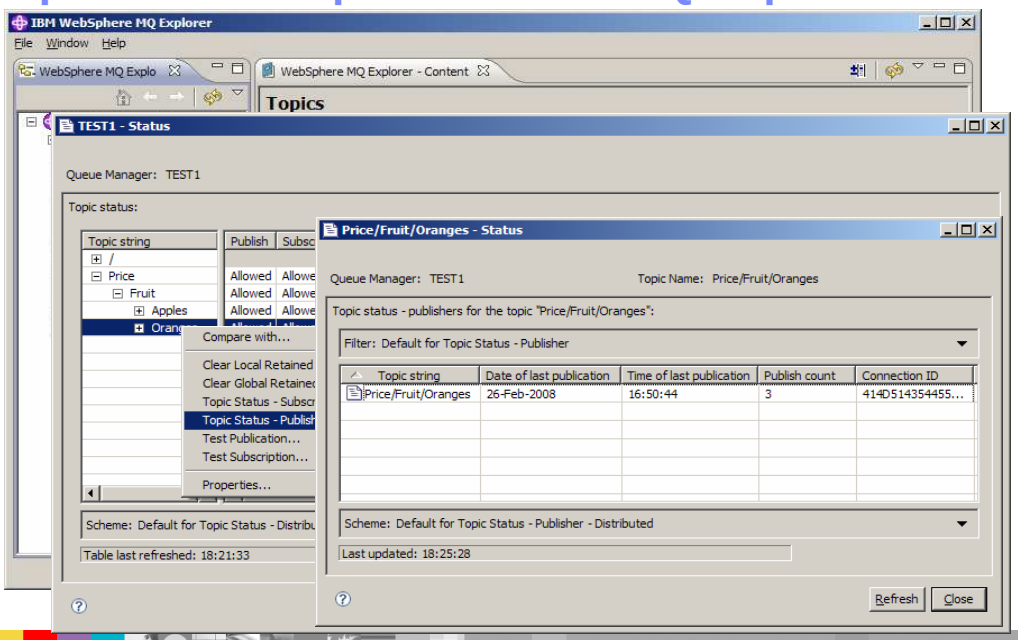
DISPLAY TPSTATUS TYPE(PUB)

```
DIS TPSTATUS( Price/Fruit/+ ) TYPE(PUB) all
AMQ8754: Display topic status details.
   TOPICSTR(Price/Fruit/Oranges)        LPUBDATE(2008 02 26)
   LPUBTIME(16:50:44)
   ACTCONN(414D5143544553543120202020202020832AC44720005E02)
   NUMPUBS(3)
AMQ8754: Display topic status details.
   TOPICSTR(Price/Fruit/Apples)         LPUBDATE(2008 02 26)
   LPUBTIME(16:50:37)
   ACTCONN(414D5143544553543120202020202020832AC44720007601)
   NUMPUBS(1)

DIS CONN(832AC44720007601) TYPE(ALL)
AMQ8276: Display Connection details.
   CONN(832AC44720007601)
   EXTCONN(414D514354455354312020202020202020)
   TYPE(CONN)
   APPLTAG(D:\q.exe)                        APPLTYPE(USER)
   USERID(hughson)

   OBJNAME( )                               OBJTYPE(TOPIC)
   OPENOPTS(MQOO OUTPUT,MQOO FAIL IF QUIESCING)
   TOPICSTR(Price/Fruit/Apples)
```


Using DISPLAY TPSTATUS for publishers, you can see the details of the current publishers on this topic string. One of the attributes returned is the active connection ID (ACTCONN). To see the details about that specific application, use the DISPLAY CONN for that connection.

With the resolution of ASPARENT values, the object that finally resolved the value can be further up the topic tree than the point at which you are publishing.

Topic status for publishers in MQ Explorer



MQ Explorer can be used to display the topic status for publishers as shown here.

Section

Summary

This section provides a summary of WebSphere MQ V7 publish/subscribe as presented in Publish/subscribe administration part 1 and part 2.

Summary

- Topic tree administration control
  - Topic object definition
  - Topic tree behavior
  - A point for security control
- No code change publish/subscribe
  - Queue aliases pointing to topics
  - Administrative subscription commands
- Administration and monitoring
  - Updates to Display CONN
  - New status displays
    - Topic status
    - Subscription status

Administration for publish/subscribe in WebSphere MQ V7 uses the standard MQ administration interfaces.

Topic object configuration and the behavior of the topic tree is controlled in the definition and setting of attributes on the topic objects. These objects are also the point for security control.

No code changes are required to use publish/subscribe; applications written to MQPUT to a queue or MQGET from a queue can be included in a publish/subscribe environment with only administrative changes.

There are updates to DISPLAY CONN and new status displays to allow monitoring of applications using the topic tree.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_PubSub_Admin_part1.ppt

This module is also available in PDF format at: ../PubSub_Admin_part1.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM WebSphere

A current list of other IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.