MQSeries® for Compaq NonStop™ Kernel

Quick Beginnings

Version 5 Release 1

GC34-5887-00

MQSeries® for Compaq NonStop™ Kernel

Quick Beginnings

Version 5 Release 1

GC34-5887-00

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix E. Notices" on page 109.

First Edition (June 2001)

This edition applies to MQSeries for Compaq NSK, Version 5.1 and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

Welcome to MQSeries for Compaq NonStop Kernel, V5.1

MQSeries for Compaq NonStop Kernel, V5.1—also referred to in this book as MQSeries or MQSeries for Compaq NSK—is part of the MQSeries family of products.

This book is primarily for system administrators who manage the configuration and administration tasks for MQSeries. It describes MQSeries for Compaq NSK and explains how to plan for and install the product. For detailed information about using MQSeries after it has been installed, refer to the *MQSeries for Compaq NSK, V5.1 System Administration Guide*.

How this book is organized

Use Table 1 to find the information you need to get started with MQSeries for Compaq NSK.

Table 1. Getting Started Road Map

If you want to...	**Refer to...**
Learn about system requirements for installing MQSeries for Compaq NSK	"Chapter 1. Planning to install MQSeries" on page 3
Install MQSeries for Compaq NSK	"Chapter 2. Installing MQSeries for Compaq NSK V5.1" on page 9
Migrate MQSeries for Compaq NSK	"Chapter 3. Migrating to MQSeries for Compaq NSK V5.1" on page 19
Learn about new features of MQSeries for Compaq NSK	"What's new in MQSeries for Compaq NSK V5.1" on page xi
Read an introduction to MQSeries concepts	"Chapter 7. About MQSeries" on page 37
Start using command sets	"Chapter 8. Using MQSeries" on page 45
View or print online documentation	"Chapter 9. Obtaining additional information" on page 63
Contact IBM	*Readers comment form* at the back of the book

Conventions

Knowing the conventions used in this book will help you use it more efficiently.

Conventions

- **Boldface type** indicates the name of an item you need to select or the name of a command.
- *Italics type* indicates new terms, book titles, or variable information that must be replaced by an actual value.
- `Monospace type` indicates an example (such as a fictitious path or file name) or text that is displayed on the screen.

What's new in MQSeries for Compaq NSK V5.1

The following new function is described in the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

Performance

New queue server process
A new queue server process has been introduced into the queue manager, which provides message storage for one or more local queues and manages all GET and PUT operations on those queues. It provides an efficient implementation for non-persistent messaging and supports the new messaging functions for Version 5.1.

Changes in status server operation
The status server replaces the file-based approach to channel status. It supports the status information of those objects that are not local queues and provides support for efficient access to channel status information.

Non-persistent messages
You can now take advantage of the performance improvements offered by non-persistent messages.

FASTPATH binding support for trusted applications
If your application is suitable, you can connect to a queue manager using FASTPATH bindings to enjoy significant performance improvements. FASTPATH applications are restricted in certain ways and must be well behaved since this form of binding provides less protection for the critical internal data of the queue manager.

Improvements to disk storage for persistent messages
As part of the new queue server architecture, the storage of persistent messages on disk has been modified to provide enhanced performance for all sizes of message. No alternate key files are required for queue files and a new type of disk storage for very large messages has been introduced that maximizes the efficiency of storage for messages up to 100 MB in size.

Enhanced MQSeries functionality

MQSeries queue manager clusters

MQSeries queue managers can be connected to form a cluster of queue managers. Within a cluster, queue managers can make the queues they host available to every other queue manager. Any queue manager can send a message to any other queue manager in the same cluster without the need for explicit channel definitions, remote queue definitions, or transmission queues for each destination. The main benefits of MQSeries clusters are:

- Fewer system administration tasks
- Increased availability
- Workload balancing

See the *MQSeries Queue Manager Clusters* book for a complete description of this function.

MQSeries Administration Interface (MQAI)

MQSeries for Compaq NSK V5.1 now supports the MQSeries Administration Interface (MQAI), a programming interface that simplifies the use of PCF messages to configure MQSeries. For more information about the MQAI, including full command descriptions, see *MQSeries® Administration Interface Programming Guide and Reference*.

Enhanced MQI support

MQSeries for Compaq NSK V5.1 now supports advanced messaging functionality provided in Version 5.0 and Version 5.1 MQSeries releases on other platforms. This includes distribution list processing, reference messages, segmented messages and many other new options. See the *MQSeries Application Programming Guide* and the *MQSeries Application Programming Reference* for more information.

Increased size of messages and message queues

A message can be up to 100 MB in size. A message queue can be up to 4 GB.

Automatic default object creation

When you use the **crtmqm** command to create a queue manager, the system default objects are automatically created.

Controlled, synchronous shutdown of a queue manager

A new option has been added to the **endmqm** command to allow controlled, synchronous shutdown of a queue manager.

Java™ support

MQSeries for Compaq NSK V5.1 supports NonStop Server for Java, Version 1.5 and later.

C++ support

MQSeries for Compaq NSK V5.1 now supports C++ compilers, allowing applications to be coded in C++.

OSS application support

MQSeries for Compaq NSK V5.1 now works with NSK OSS applications using C, C++, Cobol and Java.

Web administration

With MQSeries for Compaq NSK V5.1, you can perform the following tasks using a Microsoft® Windows NT® system in association with an HTML browser, for example, Netscape Navigator or Microsoft Internet Explorer:

- Log on as an MQSeries Administrator
- Select a queue manager and issue MQSC commands against it
- Create, edit and delete MQSC scripts.

Improved user exit mechanism

The mechanism for binding and configuring user exit code for use with MQSeries has been considerably improved to provide an interface closer to the standard, and a common mechanism for all exits.

Intercommunications

TCP/IP

MQSeries for Compaq NSK V5.1 now permits multiple Guardian TCP/IP server processes to be used by one queue manager. This means better configurations for load balancing across network hardware, and redundancy in network connections for a queue manager and applications.

SNA

MQSeries for Compaq NSK V5.1 has an improved mechanism for managing and controlling remote initiation of channels for the SNA transport protocol. This new mechanism uses a listener process that runs under PATHWAY and is supported for both SNAX and InSession ICE products. The non-standard channel attribute AUTOSTART is no longer supported.

Channels

Channels now support *heartbeats* and the ability to transmit non-persistent messages outside of a unit of work to provide better performance.

MQSeries for Compaq NSK V5.1 now supports the optional automatic definition of channels for remotely initiated channels from other queue managers or clients.

Compaq NSK-specific ease-of-use

Compaq NSK Fix Command included with runmqsc
runmqsc now includes the Compaq NSK Fix Command facility to allow you to recall and edit MQSC commands.

Enhanced altmqfls utility
The utility **altmqfls** has been changed substantially to provide detailed administrative management of the storage options for messages.

MQMC panels
The MQMC administration panels provided as part of the queue manager PATHWAY environment have been upgraded to support the enhanced MQSeries functionality in this release.

Part 1. Installing MQSeries for Compaq NSK V5.1

Chapter 1. Planning to install MQSeries

This chapter summarizes the system requirements to run MQSeries, and the decisions you must make before installing MQSeries.

Release notes

Before installing MQSeries, we recommend that you read the release notes for the product. You will find the release notes on the product tape in the subvolume Z51S*lang* where *lang* is the four character national language name (for example ENUS). This file contains any additional information about MQSeries for Compaq NSK V5.1, including limitations, known problems and workarounds, and supersedes any corresponding information within this book.

Installing for the first time?

If you are installing MQSeries for Compaq NSK V5.1 on a new system, continue reading this chapter to determine if your system meets the software and hardware requirements, then use "Chapter 2. Installing MQSeries for Compaq NSK V5.1" on page 9 to guide you through the installation.

Migrating to a new release?

Note: You **must** already have MQSeries for Tandem NSK V2.2.0.1 on your system, with PTF U473441 installed, before you can migrate to Version 5.1.

To install the latest version, continue reading this chapter to determine if your system meets the software and hardware requirements, then use "Chapter 3. Migrating to MQSeries for Compaq NSK V5.1" on page 19 to guide you through the migration.

Hardware requirements

MQSeries for Compaq NSK V5.1 requires:

- Any of the Tandem or Compaq range of machines supported by Compaq NSK D45 or later D4x, or G06 or later G06x.
- That the hardware required for your network transport protocols be in place.

Recommended but not required. If you want single point of failure fault-tolerance, you need to provide mirrored disks for:

- TMF audit space
- Queue storage for MQSeries messages.

See “Disk storage requirements” for guidance on how to estimate how much disk space you need for each of these items.

Disk storage requirements

MQSeries for Compaq NSK V5.1 requires:
- For base code and runtime: 120 MB.
- Enough space for the TMF audit trail. For each queue manager and for each MQSeries application that uses a queue manager, you need to allow for the space used in the TMF audit trail used by MQSeries. See “Estimating size of TMF audit-trail” for guidance on how to estimate the size of the audit trail volume. For more on using Compaq NSK’s transaction manager, see the *MQSeries for Compaq NSK V5.1 System Administration Guide*.
- Enough space for MQSeries messages. By default, the queue manager creates queue data files that support up to 100 MB of data each.

Estimating size of TMF audit-trail

The amount of TMF audit-trail disk space required by MQSeries depends on the mixture of message sizes you use, and on the way that applications use MQSeries. The following information will help you decide on the minimum amount of space to plan for. Note that you need to be concerned only with persistent messages; non-persistent messages are stored in memory and use a negligible amount of TMF audit-trail space even if operated on using syncpoint options.

For small persistent messages, less than 3626 bytes of data (including any transmission queue header), the message is stored entirely in the queue file record for that message. In this case, the audit trail usage is the same as the data space (for example, to store the *before* or *after* image of the data).

For persistent messages that are stored in the queue overflow file, the first 3626 bytes of message data is stored in the queue file and the remainder in the queue overflow file. In this case, the audit trail usage is again the same as the data space.

For persistent messages that are stored in message overflow files, the first 3626 bytes of message data is stored in the queue file and the remainder is stored in the unaudited message overflow file. The audit trail usage for these messages is therefore only 3626 bytes per message.

Note that the amount of audit trail space in use at any one time by MQSeries depends upon the number and size of each persistent message that is in an uncommitted state at any moment in time. The calculation of these parameters can only be approximate and depends entirely on the application design. Use the data in this section and the knowledge of your own applications to calculate the normal and peak audit trail usage required by MQSeries.

Software requirements

MQSeries for Compaq NSK V5.1 requires:

- Tandem or Compaq NSK D45 or later D4x, G06 or later G06x operating systems, including TM/MP (TMF), ENSCRIBE and EMS
- TS/MP (PATHWAY) to match operating system
- SCF for configuration, command, and control of TCP and SNA network transports
- Either:
  - TCP/IP, installed and correctly configured

    or
  - SNAX/APC (over SNAX/XF or SNAX/APN), or Insession ICE V3.2 or later, installed and correctly configured
- For instrumentation, MEASURE
- For OSS applications, OSS
- For Java applications, NonStop Server for Java Version 1.5 or later and OSS
- At least one of the following language compilers, installed and correctly configured: C, COBOL-85, or TAL.

Supported compilers

MQSeries for Compaq NSK V5.1 supports the following compilers:

- C (native and non-native)
- C++ (native only)
- COBOL-85 (native and non-native)
- TAL (non-native)
- NonStop Java Version 1.5 or later

C, C++, COBOL-85 and TAL must be compatible with the base operating system version.

Preparing for installation

Before installing MQSeries for Compaq NSK, you must:

- Create a user ID in the MQM user group to use for the installation. This is the user ID:
  - Under which all queue managers are created and run
  - Under which all product executables (rather than applications) are run

  - By which all product data files and databases are owned.

  Note: You are highly recommended to log on as a member of the MQM group when installing the product. If you do not do this, you will receive a warning and unpredictable results may occur.

- Find out which national language to use for the installation from the supported national languages. The national language is set on a system-wide basis at the time of installation and all the queue managers in an installation must use the same national language. (However, the product can be installed multiple times with different national languages for each installation.)
- Decide the location of the installation subvolumes (ISVs) where the software is initially loaded from tape, if different from the default location ($SYSTEM).

  Note: Although $SYSTEM is the default volume for the installation, it is typically not recommended because it is a busy disk volume.

  The default installation subvolumes are:

ZMQSCONV	Data conversion tables
ZMQSEXE	Product executables
ZMQSLIB	Libraries and header files
ZMQSPAX	OSS Pax file (Optional)
ZMQSSMPL	Sample code
ZMQSSYS	Product configuration files and documents
ZMQSLIC	Translated license information and license agreement files

- The configuration of the TM/MP (TMF) product is critical to the correct operation of MQSeries queue managers. All volumes that hold queue manager data files must be TMF audited. The TMF subsystem must be configured with sufficient audit-trail space for the operation of all queue managers covered by that audit trail. See “Estimating size of TMF audit-trail” on page 4 for more information.
- SNAX, ICE, or TCP/IP must be installed and configured appropriately to use MQSeries queue managers to communicate between systems (known as distributed queuing).

What to do next

When you have finished checking that your system meets the hardware, software and disk storage requirements, and you have completed the tasks listed in “Preparing for installation” on page 5, then:

- If you are installing MQSeries for Compaq NSK V5.1 on a new system, see “Chapter 2. Installing MQSeries for Compaq NSK V5.1” on page 9 for the next step.

- If you already have MQSeries for Tandem V2.2.0.1 installed on your system and you want to migrate it by installing the latest version, see“Chapter 3. Migrating to MQSeries for Compaq NSK V5.1” on page 19 for the next step.

Chapter 2. Installing MQSeries for Compaq NSK V5.1

This chapter explains how to install MQSeries for Compaq NSK V5.1 for the first time.

Note: If you already have MQSeries for Tandem NSK Version 2.2.0.1 installed on the system, then you do not need to use this chapter, unless you want to create a new parallel installation of MQSeries for Compaq NSK V5.1 since it is possible to install MQSeries many times (independently) on a single system. Instead, follow the instructions in “Chapter 3. Migrating to MQSeries for Compaq NSK V5.1” on page 19.

Before you begin

Before installing MQSeries for Compaq NSK V5.1, make sure your system meets all hardware, software and disk storage requirements and that you have created a user ID in the MQM group. The user ID MQM.MANAGER is recommended. See “Chapter 1. Planning to install MQSeries” on page 3.

Installation procedure

To install MQSeries for Compaq NSK:

1. Log on as a user in the MQM group. If this group does not exist then create the group and appropriate users.

   Note: It is highly recommended that you are logged on as a member of the MQM group when installing MQSeries. Attempting to install MQSeries while not logged on as a member of the MQM group will produce the following warning message: *You are not logged on as a member of the MQM Group Continuing can result in unexpected behaviour. Do you wish to continue? Entering NO or QUIT will terminate this installation.* If you choose to continue, the installation may complete, but any queue managers you create may not work correctly.
2. Run the RESTORE command to restore the installation utility (**instmqm**) from tape into the installation subvolume. For example:

```
RESTORE <tape device>, $*.install.*, MAP NAMES $*.*.* to
$vol.subvol.*, NOUNLOAD, LISTALL, MYID
```

where *$vol.subvol* is the volume and subvolume where you want to restore and use the **instmqm** utility.

3. To extract the release notes or documentation from the tape before running the installation program, use the information in “Restoring the README file and user documentation” on page 15 to determine the names of the files on tape. Use an appropriate RESTORE or UNPAK command to extract these files to disk.
4. After the RESTORE command is complete, verify that there are no errors and that **instmqm** is correctly restored.
5. Run the installation utility by entering **instmqm** at the TACL prompt. (For a description of the **instmqm** command, see “instmqm (Install MQSeries for Compaq NSK)” on page 87.) The installation utility will load the remaining software from tape.

   On tape, the files are structured as follows:

$*.Z51SCONV.*	Data conversion tables
$*.Z51SSYS.*	Common product configuration files
$*.INSTALL.*	Product installation files
$*.Z51MQPAX.*	OSS files
$*.Z51SLIB.*	Library files
$*.Z51SEXE.*	Executables
$*.Z51SSMPL.*	Sample programs
$*.Z51SLIC.*	License files
$*.Z51S*lang***.***	Documents in each of the national languages. *lang* is a four character national language acronym. For example, $*.Z51SENUS.* contains the documents in US English.

Figure 1 on page 11 shows the sequence of prompts (with example responses) that appear during the installation process. In this example, use the following command to restore the installation files from $*.install subvolume on tape:

```
$AUDIT1 ZMSSYS 70>restore <tape device>,$*.install.*,map names $*.*.*
 to $audit1.zmqssys.*, nounload, listall, myid
```

Where `$audit1.zmqssys` is the subvolume from where you wish to run the INSTMQM program.

Then run the **instmqm** program.

For a PAK/archive file installation example, see “Appendix D. PAK file installation examples” on page 99.

```
$AUDIT1 ZMQSSYS 71> instmqm


IBM MQSeries for Compaq NSK, Version 5.1
Installation and License update program.


@(#) Licensed Materials -
Property of IBM  5724-A39
(C) Copyright IBM Corp. 1993, 2001 All Rights Reserved
US Government Users Restricted Rights - Use,
duplication or  disclosure restricted by GSA
ADP Schedule Contract with IBM Corp.


Product installation selected...
You may stop the installation by entering
"quit" at any prompt.
Where there is a default for a parameter, you may
select it by pressing the Enter key on its own.

Phase 1: Collection of license information

License information
Enter the system type that you are licensed for.
The following system types are recognized:
CLX/R
CLX800
K100
K120
K200
CYCLONE
CLX2000
K1000
K2000
S74
S740
S7000
S7400
K10000
K20000
S70000
S72000
S74000
OTHER
There is no default value for this parameter.
```

Figure 1. Script used for installing MQSeries for Compaq NSK (Part 1 of 4)

```
Please enter your selection:  s7000

Will you be installing from tape or from an archive file?
Enter TAPE or ARCHIVE.
The default value for this parameter is "TAPE"

Please enter the selection:     tape

Enter either a local or remote tape device name.
The device name entered will be validated by opening it.
If the device cannot be opened you will be given the
opportunity to correct the name.
There is no default value for this parameter.

Please enter the tape device:   $MARA1


Enter the name of your spooler process.(eg: $S )
Do not enter the spooler location.
The name entered will be validated by opening it.
If the spooler cannot be opened you will be given the
opportunity to correct the name.
The default value for this parameter is "$S"

Please enter the spooler name:   $s

Select the type of installation to be performed.
The following options are available:
SCRATCH     - a from scratch installation
UPGRADE     - an upgrade from the latest service
              level of MQSeries V2.2.0.1
The default value for this parameter is "SCRATCH"

Please enter the type of installation:      scratch

Enter the volume that you will use for installation.
Enter the volume name in the format "$VVVVVVV".
The default value for this parameter is "$SYSTEM"
```

Figure 1. Script used for installing MQSeries for Compaq NSK (Part 2 of 4)

```
Please enter the volume:      $audit1

Enter the default volume that you want Queue Managers to
be created on.
Note that the default Queue Manager volume may be changed
at any time after installation by editing the MQSINI file.
Enter the volume name in the format "$VVVVVVV".
The default value for this parameter is "$AUDIT1"

Please enter the volume:

Enter the subvolume on AUDIT1 that you will use for executables.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSEXE"

Please enter the subvolume:

Do you wish to install the OSS pax files?
Enter YES or NO.
The default value for this parameter is "NO"

Please enter your choice:      yes

Enter the subvolume on $AUDIT1 where you want the PAX Files put.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSPAX"
```

Figure 1. Script used for installing MQSeries for Compaq NSK (Part 3 of 4)

```
Please enter the subvolume:

Select the language to be used for administration messages.
The following languages are available:
ENUS US English
ESES Spanish
FRFR French
DEDE German
ITIT Italian
JAJP Japanese
KOKR Korean
PTBR Brazilian Portuguese
ZHCN Simplified Chinese
ZHTW Traditional Chinese
The default value for this parameter is "ENUS"

Please enter the language:               enus

License verified.

You have selected the following parameters for installation:
Name of tape device:                     $MARA1
Spooler name:                            $S
Volume for installation:                 $AUDIT1
Default Queue Manager volume:            $AUDIT1
Subvolume for executables:               ZMQSEXE
Language for messages:                   ENUS
Install OSS PAX files to:                ZMQSPAX
This is not an upgrade to a prior V2.2.0.1 installation.
Beginning to restore files to $AUDIT1.
Verify that the installation media is present and online
in device $MARA1.  Enter "YES" when ready.
Ready to restore? (yes or quit):    yes

Restoring product to $AUDIT1...
Finished restoring files.
If the summary information indicates a potential error,
review the 2 spooler jobs named #instmqm, and if necessary, repeat
the installation.
Relinking native executables...
Securing files...
Creating MQSINI file...
Finished creating MQSINI file.
Creating message file...
Finished creating message file.
Installation complete.
```

Figure 1. Script used for installing MQSeries for Compaq NSK (Part 4 of 4)

Restoring the README file and user documentation

The following documents exist in each of the Z51S*lang* subvolumes on the tape. *lang* is the four letter acronym for the language. (For example, subvolume Z51SENUS contains the documents for U.S. English.)
- README
- MQSQBPDF
- MQSSAPDF

The README file is a text file that can be viewed from TACL. MQSQBPDF (*MQSeries for Compaq NSK Quick Beginnings*) and MQSSAPDF (*MQSeries for Compaq NSK System Administration Guide*) are Adobe Acrobat files (.pdf). You will need to FTP these files in binary format to a PC that has Adobe Acrobat Reader installed. If, after you install, you want to obtain these files in another language you can use RESTORE or UNPAK to retrieve them as follows:

```
RESTORE <tapedrive>,$*.Z51Slang.*,MAP NAMES $*.*.* to $vol.subvol.*,
 NOUNLOAD,LISTALL,MYID
```

where *lang* is the acronym of the national language of the documents you want to retrieve.

These files are installed in the ZMQSSYS subvolume by default during the installation.

What to do next

When you have finished installing MQSeries for Compaq NSK V5.1, you can:
- Set the environment variables as described in “Setting Compaq NSK environment variables (PARAMS)” on page 16.
- Make sure SNAX, ICE, or TCP/IP are installed and configured appropriately to use MQSeries queue managers to communicate between systems (known as distributed queuing).
- Configure the RDF environment if the queue manager is to be used in an RDF environment. See “Configuring RDF for MQSeries for Compaq NSK” on page 16 for more information.
- Move the required files into the OSS file system, if you elected to install the OSS PAX files option when you installed MQSeries. See “Installing OSS components” on page 16 for more information.

When those tasks are complete, you can verify that the installation was successful by following the procedure outlined in “Chapter 4. Verifying the installation for MQSeries for Compaq NSK V5.1” on page 27.

Setting Compaq NSK environment variables (PARAMS)

MQSeries for Compaq NSK creates and uses some Compaq NSK environment variables (PARAMS). These PARAMS are listed in “Appendix C. Setting TACL environment variables for MQSeries for Compaq NSK” on page 95. Before you verify the installation, you must set the MQDEFAULTPREFIX PARAM to ensure that MQSeries can find the installed files. You can set the other PARAMS later, when it is convenient for you.

To set the MQDEFAULTPREFIX PARAM now:

1. At a TACL prompt, type:

```
PARAM MQDEFAULTPREFIX $data00
```

   where `$data00` is the volume where MQSeries was installed.

Although it is not required to verify the installation, we recommend that you eventually:

- Include all the MQSeries PARAM statements in your TACLCSTM files so that when you log on, these PARAMS are set automatically and correctly and any programs run from the TACL inherit the correct values.
- Set the PMSEARCH environment variable so that the system can find MQSeries executables.

You can do this now or as part of your set up. For more on setting the PMSEARCH and MQSeries PARAMs, see “Appendix C. Setting TACL environment variables for MQSeries for Compaq NSK” on page 95.

Configuring RDF for MQSeries for Compaq NSK

If the queue manager is to be used in an RDF environment, you must configure RDF to operate on all volumes used for the queue manager. Take particular care if queues have been moved to alternative volumes. RDF must be configured with identically named volumes on both the primary and backup sites.

Installing OSS components

If you elected to install the OSS PAX files when installing MQSeries, you need to move the required files into the OSS file system. To do this:

1. Change to the subvolume in which you elected to place the OSS files. In this subvolume there are two files:

  - The `README` file gives detailed instructions on Java and configuring MQSeries in the OSS environment.
  - The file `MQMPAX` is an OSS packed file that contains all the OSS files.
2. Install the required files onto the OSS file system by issuing the following command at a TACL prompt:

```
PINSTALL -rvf/G/vol/subvol/mqmpax
```

   where `vol` and `subvol` are the volume and subvolume you selected to install the OSS files.

The OSS files are extracted to the `/opt/mqm` directory on your OSS file system.

Chapter 3. Migrating to MQSeries for Compaq NSK V5.1

This chapter shows you how to migrate (upgrade) from MQSeries for Tandem NSK Version 2.2.0.1 to MQSeries for Compaq NSK V5.1.

Note: If you are installing MQSeries for Compaq NSK V5.1 on a system that does not have any of the previous versions of MQSeries installed, then use the procedure described in "Chapter 2. Installing MQSeries for Compaq NSK V5.1" on page 9.

To migrate to the latest version, you:

- Perform the tasks described in "Before you begin". This includes stopping all queue managers, making sure the latest maintenance fix is installed on the system, and backing up your system.
- Install the latest version and then relink applications. This is described in "Migrating to MQSeries for Compaq NSK V5.1" on page 20.
- Run the **upgmqm** utility for each queue manager that needs migrating.
- Check that the migration has worked properly by running the verification procedure described in "Chapter 4. Verifying the installation for MQSeries for Compaq NSK V5.1" on page 27.

Before you begin

Before you begin upgrading to MQSeries for Compaq NSK V5.1:

- Stop all queue managers. Use the **endmqm** command. See "endmqm (End queue manager)" on page 84 for more information.
- Take a backup of all your current MQSeries software, databases, queue managers and application programs that are linked with MQSeries. Ensure that your backup includes any queue files that have been moved from their default locations.
- Check that your system meets all the requirements described in "Chapter 1. Planning to install MQSeries" on page 3, and that you have considered all the points outlined in "Preparing for installation" on page 5.
- Make sure that you are currently running MQSeries for Tandem V2.2.0.1, that all your applications are working at this level and that you have installed the latest maintenance fix (PTF U473441). If you are not sure what is running on your system, see "Querying the service level" on page 20 for guidance on how to find out.

Querying the service level

To find out the current level of MQSeries installed on your Compaq NSK system, read the ZMQSSYS.MEMOPTF file and look for the first entry of CSD history. The line should contain the text `PTF U473441`. If the PTF number is different, then you do not have the latest PTF installed.

In addition to checking the MEMOPTF file, run VPROC on the MQSRLLIB object code (in the ZMQSSYS subvolume). A binder timestamp of `12OCT2000 03:10:30` or later indicates that PTF U473441 is installed.

Migrating to MQSeries for Compaq NSK V5.1

To migrate from MQSeries for Tandem NSK Version 2.2.0.1 to MQSeries for Compaq NSK V5.1:

1. Log on as the user in the MQM group who performed the Version 2.2.0.1 installation. (This user was MQM.MANAGER by default in Version 2.2.0.1.) If you do not have a user ID in the MQM group, then create one.
2. Run the RESTORE command to restore the installation utility (**instmqm**) from tape into the installation subvolume. For example:

   ```
   RESTORE <tape device>, $*.install.*, MAP NAMES
    $*.*.* to $vol.subvol.*, NOUNLOAD, LISTALL, MYID
   ```

   where *$vol.subvol* is the volume and subvolume where you want to restore and use the **instmqm** utility.
3. After the RESTORE command is complete, verify that there are no errors and that **instmqm** is correctly restored.
4. Run the installation utility by entering **instmqm** at the TACL prompt. (For a description of the **instmqm** command, see “instmqm (Install MQSeries for Compaq NSK)” on page 87.) The installation utility loads the remaining software from tape.
5. A sequence of prompts appear during the installation process. When you are asked to select the type of installation, select the UPGRADE option. Here is an example of the upgrade installation:

```
$AUDIT1 INSTALL 79> instmqm
$AUDIT1 INSTALL 79..


IBM MQSeries for Compaq NSK, Version 5.1
Installation and License update program.


@(#) Licensed Materials - Property of IBM 5724-A39
(C) Copyright IBM Corp. 1993, 2001 All Rights Reserved
US Government Users Restricted Rights - Use, duplication
or  disclosure restricted by GSA ADP Schedule Contract with IBM Corp.


Product installation selected...
You may stop the installation by entering
"quit" at any prompt.
Where there is a default for a parameter, you may
select it by pressing the Enter key on its own.

Phase 1: Collection of license information

License information
Enter the system type that you are licensed for.
The following system types are recognized:
CLX/R
CLX800
K100
K120
K200
CYCLONE
CLX2000
K1000
K2000
S74
S740
S7000
S7400
K10000
K20000
S70000
S72000
S74000
OTHER
There is no default value for this parameter.
```

Figure 2. Script used for migrating toMQSeries for Compaq NSK (Part 1 of 4)

```
Please enter your selection:  s7000

Will you be installing from tape or from an archive file?
Enter TAPE or ARCHIVE.
The default value for this parameter is "TAPE"

Please enter the selection:      tape

Enter either a local or remote tape device name.
The device name entered will be validated by opening it.
If the device cannot be opened you will be given the
opportunity to correct the name.
There is no default value for this parameter.

Please enter the tape device:      $MARA1

Enter the name of your spooler process.(eg: $S )
Do not enter the spooler location.
The name entered will be validated by opening it.
If the spooler cannot be opened you will be given the
opportunity to correct the name.
The default value for this parameter is "$S"

Please enter the spooler name:      $s

Select the type of installation to be performed.
The following options are available:
SCRATCH     - a from scratch installation
UPGRADE     - an upgrade from the latest service
              level of MQSeries V2.2.0.1
The default value for this parameter is "SCRATCH"

Please enter the type of installation:      upgrade

Enter the volume that you installed MQSeries on.
Enter the volume name in the format "$VVVVVVV".
The default value for this parameter is "$SYSTEM"
```

Figure 2. Script used for migrating toMQSeries for Compaq NSK (Part 2 of 4)

```
Please enter the volume:      $audit1

Indicate the name of the MQSeries Initialization file.
This file is usually called MQSINI
Enter this in the format "$Vol.Subvol.Filename".
The default value for this parameter is "$AUDIT1.ZMQSSYS.MQSINI".
Please enter the fully qualified name of the MQSeries Initialization file.
$audit1.zmqssys.mqsini

Enter the subvolume on $AUDIT1 containing the MQSeries executables.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSEXE"

Please enter the subvolume:   $audit1


Verifying latest service level of V2.2.0.1 is present...

Presence of PTF U473441 has been verified.
Installation proceeding.
Do you wish to install the OSS pax files?
Enter YES or NO.
The default value for this parameter is "NO"

Please enter your choice:      yes

Enter the subvolume on $AUDIT1 where you want the PAX Files put.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSPAX"

Please enter the subvolume:

Select the language to be used for administration messages.
The following languages are available:
ENUS US English
ESES Spanish
FRFR French
DEDE German
ITIT Italian
JAJP Japanese
KOKR Korean
PTBR Brazilian Portuguese
ZHCN Simplified Chinese
ZHTW Traditional Chinese
The default value for this parameter is "ENUS"
```

Figure 2. Script used for migrating toMQSeries for Compaq NSK (Part 3 of 4)

```
Please enter the language:     enus

License information updated successfully
License verified.

Tape device for installation:           $MARA1
Spooler name:                           $S
Volume for installation:                $AUDIT1
Subvolume for executables:              $AUDIT1
Language for messages:                  ENUS
Install OSS PAX files to:               ZMQSPAX
This is an upgrade to a prior V2.2.0.1 installation.
The existing MQSeries Initialization file is $AUDIT1.ZMQSSYS.MQSINI
Beginning to restore files to $AUDIT1.
Verify that the installation media is present and online
in device $MARA1.  Enter "YES" when ready.
Ready to restore? (yes or quit):    yes

Restoring product to $AUDIT1...
Finished restoring files.
If the summary information indicates a potential error,
review the 2 spooler jobs named #instmqm, and if necessary, repeat
the installation.
Relinking native executables...
Securing files...
Finished securing files.
Updating MQSINI file =$AUDIT1.ZMQSSYS.MQSINI...
Finished updating MQSINI file $AUDIT1.ZMQSSYS.MQSINI
Creating message file...
Finished creating message file.
Installation complete.
```

Figure 2. Script used for migrating toMQSeries for Compaq NSK (Part 4 of 4)

6. The migration is complete. See “Migrating message files, configuration files and applications” for descriptions of how to migrate applications and messages.

Migrating message files, configuration files and applications

To continue migrating from V2.2.0.1 to MQSeries for Compaq NSK V5.1, you need to:

- Upgrade individual queue managers using the **upgmqm** command. The utility invoked by **upgmqm** sends progress messages to the terminal from which it is started. When it has completed, the queue manager is ready to be used with this version. New Version 5.1 attributes are set to their default values. You can alter these in the usual way. For more information about **upgmqm**, see “upgmqm (Upgrade V2.2.0.1 queue manager)” on page 93.
- Recompile and rebind all Version 2.2.0.1 applications with Version 5.1 header files and libraries.

Restoring the previous backup version

If you experience difficulties with the new MQSeries Version 5.1 environment and your existing applications, we recommend that you revert to the previous version of MQSeries by restoring your previous backup.

To reinstall the backup version of MQSeries for Compaq NSK Version 2.2.0.1:

1. Delete all migrated configuration files, message files and applications.
2. Run the RESTORE command to restore the backup version. For example:

```
RESTORE <tape device>, $*.zmqs*.*,  MAP NAMES $*.*.*
 to $vol.*.*, NOUNLOAD, LISTALL, MYID
```

   where *$vol* is the volume where you want to restore the MQSeries system.

   Note: In this example, the RESTORE command would only restore the MQSeries product files (programs and configuration files). To restore backed-up queue managers, you will need to use a RESTORE command that includes all subvolumes and files that were backed up and ensure that the location they were restored to is the same as before the migration attempt was made.

Chapter 4. Verifying the installation for MQSeries for Compaq NSK V5.1

After you have installed MQSeries for Compaq NSK V5.1 and its samples components, you can use the following procedure to verify that the installation was successful.

When you have finished the verification test, you can use the steps in "Cleaning up after the verification procedure" on page 29 to remove the queue manager and leave a 'clean' system.

Before you begin

Before you begin the verification procedure:

- Make sure you are logged in as a member of the MQM group.
- Make sure the home terminal is paused.
- Make sure TM/MP (TMF) is running.
- You are advised to set the environment variable PMSEARCH.
- Make sure PARAM MQDEFAULTPREFIX is set.

Verification procedure

To verify that MQSeries for Compaq NSK V5.1 installed correctly, you can create a queue manager using the **crtmqm** command. The following instructions use the name QMNAME for the queue manager. When creating your own queue manager, replace each occurrence of QMNAME with the appropriate name.

Note: A queue manager name must be unique within your network. As well, all commands and object names and most parameters in these steps are case sensitive.

1. Create a queue manager called QMNAME using the **crtmqm** command. For example, type:

```
crtmqm -n $QMNA -o $TRMG.#A -s $QMSS -v $M01Q QMNAME
```

The **crtmqm** command requires the process names of a PATHMON process, a default status server process, and a default queue server process to use for the queue manager. These process names must be unique on the system.

It also requires the name of a home terminal, which must be paused.

You must enter the following options before the name of the queue manager:

– n PATHMON process name
– o home terminal (must be paused)
– s Status server process name
– v Queue server process name

For a detailed description of the **crtmqm** command and options see “crtmqm (Create queue manager)” on page 77.

2. Start the queue manager using the **strmqm** command. For example, type:

```
strmqm QMNAME
```

The **strmqm** command returns control when the queue manager has started and is ready to accept connect requests.

3. Enable the MQSC commands by typing:

```
runmqsc QMNAME
```

4. Define a local queue (in this example, called ORANGE.LOCAL.QUEUE).

```
DEFINE QLOCAL (ORANGE.LOCAL.QUEUE)
```

You have now defined a default queue manager called QMNAME and queue called ORANGE.LOCAL.QUEUE.

5. Exit from RUNMQSC.

To test the queue and queue manager, use the samples **amqsput** (to put a message on the queue) and **amqsget** (to get the message from the queue).

1. Type the following command:

```
AMQSPUT ORANGE.LOCAL.QUEUE QMNAME
```

2. Type some message text and the press Enter.

   Your message is now in the queue and the command prompt is displayed.
3. To get the message from the queue, type:

```
AMQSGET ORANGE.LOCAL.QUEUE QMNAME
```

   The sample program starts, your message is displayed, the sample ends and the command prompt is displayed.

The verification is complete.

Cleaning up after the verification procedure

Note: Deleting the queue manager does not delete the installation. You can use this procedure even if you have previously performed it.

1. Stop the queue manager using the command:

```
endmqm QMNAME
```

2. Delete the queue manager using the command:

```
dltmqm QMNAME
```

   This command deletes the queue manager and its associated objects.

Chapter 5. Troubleshooting

If your installation was not successful or commands failed to run properly, consider the following:

- *Did you enter the commands correctly?*

  Try running one or more of the commands again. These commands and most parameters are case sensitive. If you create a queue manager with an upper case name, you must specify the name in upper case on any commands referring to that queue manager. For example, if you create a queue manager called QMNAME, you cannot use 'qmname' or 'QMname'.

  If you type in the name of a queue manager that does not exist, the message `MQSeries queue manager does not exist` is displayed.
- *Do you have enough disk space or memory to run the verification?*

  Check any error messages for an indication. If error message `AMQ7065 Insufficient space on disk` is returned, use the *dsap* utility to display the free space on the target volume. If there is insufficient free space, choose a different volume on which to create the queue manager, or free some space on the existing volume.
- *Do you have the required authority to run the commands?*

  Check that you are still logged in as a member of the MQM group and have a valid principal defined.
- *Is the home terminal you specified on the crtmqm command correct?*
- *Is the home terminal paused?*
- *Are the PATHMON, default status server and queue server process names you specified for the* **crtmqm** *command unique in your system?*

For more troubleshooting tips, see the Problem Determination chapter in the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

Chapter 6. Removing MQSeries

Before you remove MQSeries, do the following:

- Stop all MQSeries applications.
- Stop all the channels and end all queue managers cleanly using the **endmqm** command.
- Delete the queue managers using the **dltmqm** command.

To remove the MQSeries product from your Compaq NSK system, delete all of the files in the `ZMQS*` subvolumes, using the **purge** command.

Part 2. Getting started with MQSeries

Chapter 7. About MQSeries

This chapter introduces IBM® MQSeries. It describes its basic functions and its relationships with operating systems, applications, and other middleware products. It contains the following sections:

- "Introduction"
- "Messages, queues, and queue managers" on page 38
- "MQSeries configurations" on page 39
- "MQSeries capabilities" on page 42
- "Programming MQSeries" on page 44

Introduction

MQSeries is a communications system that provides assured, asynchronous, once-only delivery of data across a broad range of hardware and software platforms.

These characteristics make MQSeries the ideal infrastructure for application-to-application communication, and make it an appropriate solution whether the applications run on the same machine or on different machines that are separated by one or more networks.

MQSeries supports all the important communication protocols and even provides routes between networks that use different protocols. MQSeries bridges and gateway products allow easy access (with little or no programming) to many existing systems and application environments—for example, Lotus Notes®, Web browsers, Java applets, and many others.

The assured delivery capability reflects the many functions built in to MQSeries to ensure that data is not lost because of failures in the underlying system or network infrastructure. Assured delivery enables MQSeries to form the backbone of critical communication systems and to be entrusted with delivering high-value data. There are also options that allow you to select a less robust quality of service, where this is appropriate. For example, there might be circumstances where you might prefer faster delivery with less emphasis on assured delivery.

The asynchronous processing support in MQSeries means that the exchange of data between the sending and receiving applications is time independent. This allows the sending and receiving applications to be decoupled so that the sender can continue processing, without having to wait for the receiver to acknowledge that it has received the data. In fact, the target application does not even have to be running when the data is sent. Likewise, the entire network path between the sender and receiver may not need to be available when the data is in transit.

Once-only delivery of data is a vital consideration, particularly in financial and business applications where duplicate requests to move large sums of money from one account to another are precisely what you do not want to happen!

Messages, queues, and queue managers

The three fundamental concepts in MQSeries that you need to understand are:
- Messages
- Queues
- Queue managers

Messages

A *message* is a string of bytes that has meaning to the applications that use it. Messages are used for transferring data from one application to another (or to different parts of the same application). The applications can be running on the same platform, or on different platforms.

MQSeries messages have two parts; the *application data* and a *message descriptor*. The content and structure of the application data is defined by the application programs that use the data. The message descriptor identifies the message and contains other control information, such as the type of message and the priority assigned to the message by the sending application.

Queues

A *queue* is a data structure in which messages are stored. The messages may be put on, or got from, the queue by applications or by a queue manager as part of its normal operation.

Queues exist independently of the applications that use them. A queue can exist in main storage (if it is temporary), on disk or similar auxiliary storage (if it must be kept in case of recovery), or in both places (if it is currently being used, and must also be kept for recovery). Each queue belongs to a *queue manager*, which is responsible for maintaining it. The queue manager puts the messages it receives onto the appropriate queue.

Queues can exist either in your local system, in which case they are called *local queues*, or at another queue manager, in which case they are called *remote queues*.

Applications send to, and receive messages from, queues. For example, one application can put a message on a queue, and another application can get the message from the same queue.

Each queue has *queue attributes* that determine what happens when applications reference the queue. The attributes indicate:

- Whether applications can retrieve messages from the queue (get enabled)
- Whether applications can put messages onto the queue (put enabled)
- Whether access to the queue is exclusive to one application or shared between applications
- The maximum number of messages that can be stored on the queue at the same time (maximum queue depth)
- The maximum size of messages that can be put on the queue (maximum message size)

Queue managers

A queue manager provides queuing services to applications, and manages the queues that belong to it. It ensures that:

- Object attributes are changed according to the details received.
- Special events (such as instrumentation events or triggering) are generated when the appropriate conditions are met.
- Messages are put on the correct queue, as requested by the application. The application is informed if this cannot be done, and an appropriate reason code is given.

Each queue belongs to a single queue manager and is said to be a *local queue* to that queue manager. The queue manager to which an application is connected is said to be the local queue manager for that application. For the application, the queues that belong to its local queue manager are local queues. A *remote queue* is a queue that belongs to another queue manager. A *remote queue manager* is any queue manager other than the local queue manager. A remote queue manager may exist on a remote machine across the network or it may exist on the same machine as the local queue manager. MQSeries supports multiple queue managers on the same machine.

MQSeries configurations

In the simplest configurations, MQSeries is installed on a machine and a single queue manager is created. This queue manager then allows you to define queues. Local applications can then use these queues to exchange messages.

Communication by applications with queues managed by another queue manager requires *message channels* to be defined. It is not necessary to define a channel directly to the target queue manager and it is often appropriate to define one only to the next hop (that is, an intermediate queue manager). Message channels available from that queue manager will be used to deliver the message to the target queue manager (or even to a subsequent hop).

More complex configurations can be created using a client-server structure. The MQSeries product can act as an MQSeries server to MQSeries clients. The clients and server do not need to be on the same platform. MQSeries supports a broad range of client platforms. The MQSeries products typically include clients for a variety of platforms. Additional MQSeries clients are available from the MQSeries Web site.

In a client-server configuration, the MQSeries server provides messaging and queuing services to the clients, as well as to any local applications. The clients are connected to the server through dedicated channels (known as *client channels*) for clients. This is a cost-effective deployment method because a server can support hundreds of clients with only a single copy of the MQSeries server product. However, the client channel must be continuously available whenever the MQSeries applications on the client are running. This contrasts with the message channels, which need not be continuously available to support MQSeries applications running on the server.

See “Channels” for more information.

MQSeries also supports the concept of *clusters* to simplify setup and operation. A cluster is a named collection of queue managers and any one queue manager can belong to none, one, or several such clusters. The queue managers in a cluster can exist on the same or different machines.

There are two major benefits from the use of clusters:

1. Communication between members of a cluster is greatly simplified, particularly because the channels required for exchanging messages are automatically defined and created as needed.
2. Some or all of the queues of participating queue managers can be defined as being cluster queues, which has the effect of making them automatically known and available to all other queue managers in the cluster.

See “Clusters” on page 41 for more information.

Channels

A channel provides a communication path to a queue manager. There are two types of channel: message channels and MQI channels.

A *message channel* provides a communication path between two queue managers on the same, or different, platforms. The message channel is used for transmitting messages from one queue manager to another, and shields the application programs from the complexities of the underlying networking protocols. A message channel can transmit messages in one direction only. Two message channels are required if two-way communication is required between two queue managers.

A *client channel* (also known as an *MQI channel*) connects an MQSeries client to a queue manager on a server machine and is bidirectional.

If you want to read more information about channels and how MQSeries uses them to communicate across the systems in your network, see the *MQSeries Intercommunication* book.

Clients and servers

MQSeries supports client-server configurations for MQSeries applications.

An *MQSeries client* is a part of the MQSeries product that is installed on a machine to accept MQSeries calls from applications and pass them to an *MQSeries server* machine. There they are processed by a queue manager. Typically, the client and server reside on different machines, but they can also exist on the same machine.

An *MQSeries server* is a queue manager that provides queuing services to one or more clients. All the MQSeries objects (for example, queues) exist only on the queue manager machine (that is, on the MQSeries server machine). A server can support local MQSeries applications as well.

The difference between an MQSeries server and an ordinary queue manager is that the MQSeries server can support MQSeries clients, and each MQSeries client application has a dedicated communication link with the MQSeries server.

Note: MQSeries for Compaq NSK runs as a server only. There are no MQSeries clients for Compaq NSK.

For more information about client support, see the *MQSeries Clients* book.

Clusters

A cluster is a named collection of queue managers.

Clusters require that at least one of the queue managers in the cluster be defined as a *repository* (that is, a place where the shared cluster information can be held). More typically, two or more such repositories are usually designated to provide continued availability in the case of system failure. MQSeries makes sure that the information in the repositories is synchronized.

MQSeries configurations

Because of the fault tolerant design of the Compaq NSK platform, it can serve as a highly reliable repository for a cluster.

When a queue is defined as a cluster queue, it can be regarded as a public queue in that it is freely available to other queue managers in the cluster. This contrasts with non-cluster queues, which are accessible only when a local definition of them is available. Thus, a non-cluster queue has the characteristics of a private queue, accessible only to those queue managers that have been configured to know about them.

Public queues with the same name in the same cluster are regarded as equivalent. If a message is sent to that queue name, MQSeries (by default) sends it to any one of the instances, using a load-balancing algorithm. If you do not want this to happen, you can use the queue manager and queue name in the address, thus forcing the message to be delivered to a specific queue manager. Alternatively, you can replace the load-balancing routine with a different implementation. This is typical of MQSeries, in that there are many examples of where standard behavior can be changed by implementing user code in exits designed for this purpose.

You can read a full explanation in the *MQSeries Queue Manager Clusters* book.

MQSeries capabilities

MQSeries can be used to create many different types of solutions. Some exploit the platform support, or the bridge and gateway capabilities, to connect existing systems in an integrated way or to allow new applications to extract information from, or interchange information with, existing systems. Other solutions support business application servers, where a central pool of MQSeries applications can manage work sent across networks. Complex routing of information for workflow scenarios can be supported. Publish/subscribe or “send and forget” are other application scenarios that use different message flows. Load balancing and hot-standby systems can be built using the power and flexibility of MQSeries, which includes specific functions to support many of these diverse scenarios.

See the *MQSeries Application Programming Guide* for more information about writing MQSeries applications.

Transactional support

An application program may need to group a set of updates into a *unit of work*. Such updates are usually logically related and must all be successful for data integrity to be preserved. Data integrity would be lost if one update in the group succeeded while another failed.

Note: MQSeries as a family, supports transactional messaging, however transactional support in MQSeries for Compaq NSK relies on Compaq NSK's own TM/MP to maintain transactional integrity.

A unit of work *commits* when it completes successfully. At this point all updates made within that unit of work are made permanent and irreversible. Alternatively, all updates are *backed out* if the unit of work fails. *Syncpoint coordination* is the process by which a unit of work is either committed or backed out with integrity.

Commit and back out are provided as part of the TM/MP Transaction environment on Compaq NSK.

Instrumentation events

You can use MQSeries instrumentation events to monitor the operation of queue managers.

Instrumentation events cause special messages, called *event messages*, to be generated whenever the queue manager detects a predefined set of conditions. For example, a *Queue Full* event message is generated if: Queue Full events are enabled for a specified queue; an application issues an MQPUT call to put a message on that queue; and the call fails because the queue is full.

Other conditions that can give rise to instrumentation events include:
- A predefined limit for the number of messages on a queue being reached
- A queue not being serviced within a specified time
- A channel instance being started or stopped

If you define your event queues as remote queues, you can put all the event queues on a single queue manager (for those nodes that support instrumentation events). You can then use the events generated to monitor a network of queue managers from a single node.

MQSeries instrumentation events are categorized as follows:

Queue manager events
These are related to the definitions of resources within queue managers. For example, if an application attempts to open a queue but the associated user ID is not authorized to perform that operation, a queue manager event is generated.

Performance events
These are notifications that a threshold condition has been reached by a resource. For example, a queue depth limit has been reached or, following an MQGET request, a queue has not been serviced within a predefined period of time.

Channel events

These are reported by channels as a result of conditions detected during their operation. For example, a channel event is generated when a channel instance is stopped.

Message-driven processing

When they arrive on a queue, messages can automatically start an application, using a mechanism known as *triggering*. If necessary, the application can be stopped when the message or messages have been processed.

Programming MQSeries

MQSeries applications can be developed using a variety of programming languages and styles. Procedural and object-oriented programming is supported, depending on the MQSeries platform, using, for example, Visual Basic, C, C++, Java, COBOL, PL/I, and TAL.

MQSeries function is logically divided into what is normally required by applications (such as putting messages on a queue) and what is necessary for administration (such as changing queue or queue manager definitions). Application function is known as the *MQI* (message queue interface). Administration function is known as the *MQAI* (message queuing administration interface). Applications can mix MQI and MQAI functionality, as required.

The administration functions can be implemented in two ways:

1. Most often, using MQAI language bindings
2. Sending messages to administration queues, to achieve the same results as with the MQAI, using programmable command formats (PCFs)

Chapter 8. Using MQSeries

This chapter introduces the command sets that can be used to perform system administration tasks on MQSeries objects. It covers:

Administration tasks include creating, starting, altering, viewing, stopping, and deleting MQSeries objects such as queue managers, queues, processes, channels, and namelists. To perform these tasks, you must select the appropriate command from one of the supplied command sets.

Introducing command sets

MQSeries provides the following command sets for performing administration tasks:

- Control commands
- MQSC commands
- PCF commands
- MQAI (MQSeries Administrator Interface)

This section describes the command sets that are available. Some tasks can be performed using either a control command or an MQSC command, but other tasks can be performed using only one type of command. For a comparison of the facilities provided by the different types of command set, see the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

In addition:

- Some TS/MP (PATHWAY) commands are used for administration purposes.
- The MQM (Message Queue Management) facility supports some administration tasks. The MQM is described in the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

This chapter introduces the MQSC, PCF, and control command sets, and provides a summary of the functions supported by each command set in the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

Control commands

Control commands fall into three categories:

- *Queue manager commands*, including commands for creating, starting, stopping, and deleting queue managers and command servers.
- *Channel commands*, including commands for starting and ending channels and channel initiators.
- *Utility commands*, including commands associated with authority management and conversion exits.

Using control commands

You run control commands from the TACL prompt. Command names are not case sensitive. (Note, however, that queue manager names *are* case sensitive.)

Note: If aliases are set up to use MQSeries commands from OSS, they are case sensitive.

For example:

```
runmqsc
```

MQSeries for Compaq NSK V5.1 System Administration Guide explains the syntax and purpose of each command.

The following table contains a brief description of each of the control commands. You can obtain help for the syntax of any of the commands by entering the command with -? or -h as the only parameter. MQSeries responds by listing the syntax required for the selected command.

Command	Description
altmqfls	Allows the assignment of queue and status servers, queue file location, queue server storage options and queue resize operations.

Command	Description
altmqusr	Defines or removes a principal corresponding to a Compaq NSK user ID that will have access to MQSeries.
cleanrdf	Performs routine housekeeping on the primary system queue manager in an RDF environment.
cnvclchl	Converts the client channel definition file, created for CLNTCONN channels by MQSC, from a Compaq structured file to an unstructured format acceptable to MQSeries clients.
crtmqcvx	Creates a fragment of code that performs data conversion on data type structures.
crtmqm	Creates a local queue manager and defines the default and system objects.
dltmqm	Deletes a specified queue manager.
dspmqaut	Displays the current authorizations to a specified object.
dspmqcsv	Displays the status of the command server for the specified queue manager.
dspmqfls	Displays the real file system name for all MQSeries objects that match a specified criterion.
dspmqtrc	Displays MQSeries formatted trace output.
dspmqusr	Displays details about a specified principal or all principals for the queue manager.
endmqcsv	Stops the command server on the specified queue manager.
endmqm	Stops a specified local queue manager.
endmqtrc	Ends tracing for the specified entity or all entities.
instmqm	Installs MQSeries for Compaq NSK.
runmqchi	Runs a channel initiator process.
runmqchl	Starts either a Sender (SDR) or a Requester (RQSTR) channel.
runmqdlq	Starts the dead-letter queue (DLQ) handler, a utility that you can run to monitor and handle messages on a dead-letter queue.
runmqlsr	Runs a TCP/IP listener process.
runmqsc	Issues MQSC commands to a queue manager.
runmqtrm	Invokes a trigger monitor.
setmqaut	Changes the authorizations to an object or to a class of objects.
strmqcsv	Starts the command server for the specified queue manager.
strmqm	Starts a local queue manager.
strmqtrc	Enables tracing.
upgmqm	Upgrades a V2.2.0.1 queue manager for use with MQSeries for Compaq NSK V5.1.

For more information about the syntax and purpose of control commands, see the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

MQSeries (MQSC) commands

You use the MQSeries (MQSC) commands to manage queue manager objects, including the queue manager itself, channels, queues, and process definitions. For example, there are commands to define, alter, display, and delete a specified queue.

When you display a queue, using the DISPLAY QUEUE command, you display the queue *attributes*. For example, the MAXMSGL attribute specifies the maximum length of a message that can be put on the queue. The command does not show you the messages on the queue.

For detailed information about each MQSC command, see the *MQSeries MQSC Command Reference*.

Running MQSC commands

You run MQSC commands by invoking the control command **runmqsc**. You can run MQSC commands:

- Interactively by typing them at the keyboard
- As a sequence of commands from a text file.

For more information about using MQSC commands, see the *MQSeries System Administration* book.

PCF commands

MQSeries programmable command format (PCF) commands allow administration tasks to be programmed into an administration program. In this way you can create queues and process definitions, and change queue managers, from a program. PCF commands cover the same range of functions that are provided by the MQSC facility. You can therefore write a program to issue PCF commands to any queue manager in the network from a single node. In this way, you can both centralize and automate administration tasks.

Note: Unlike MQSC commands, PCF commands and their replies are not in a text format that you can read.

For a complete description of the PCF data structures and how to implement them, see the *MQSeries Programmable System Management* book.

MQAI commands

The MQAI is a programming interface to MQSeries, using the C language. It performs administration tasks on an MQSeries queue manager using *data bags*.

Data bags allow you to handle properties (or parameters) of objects in a way that is easier than using the other administration interface, Programmable Command Formats (PCFs).

The MQAI offers easier manipulation of PCFs than using the MQGET and MQPUT calls. You can use the MQAI to:

- Implement self-administering applications and administration tools.
- Simplify the use of PCF messages. The MQAI is an easy way to administer MQSeries; you do not have to write your own PCF messages and thus avoid the problems associated with complex data structures.
- Handle error conditions more easily. It is difficult to get return codes back from the MQSeries commands (MQSC), but the MQAI makes it easier for the program to handle error conditions.

Note: MQSeries for Compaq NSK V5.1 provides C header files only. It does not provide Visual Basic header files. For more information about the MQAI, see the *MQSeries Administration Interface Programming Guide and Referenc*e.

Working with queue managers

This section describes how you can perform operations on queue managers, such as creating, starting, stopping, and deleting them. MQSeries provides control commands for performing these tasks.

Before you can do anything with messages and queues, you must create at least one queue manager.

Creating a queue manager

The following command:

- Creates a default queue manager called saturn.queue.manager
- Creates the default and system objects automatically
- Specifies the names of both a default transmission queue and a dead-letter queue.

```
crtmqm -q -d MY.DEFAULT.XMIT.QUEUE -n $MQPW -o $vhs -s $MQSS -v $MQQS
  -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.

-n Pathmon Process Name
Is the name of the process.

-o Hometerm
Is the name of the home terminal.

-s Status Server Process
Is the name of the status server process.

-v Queue Server Process
Is the name of the queue server process.

-u SYSTEM.DEAD.LETTER.QUEUE
Is the name of the dead-letter queue.

saturn.queue.manager
Is the name of this queue manager. This must be the last parameter specified on the **crtmqm** command.

For more information about these attributes, see the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

Starting a queue manager

Although you have created a queue manager, it cannot process commands or MQI calls until it has been started. Start the queue manager by typing in this command:

```
strmqm saturn.queue.manager
```

The **strmqm** command does not return control until the queue manager has started and is ready to accept connect requests.

Stopping a queue manager

To stop a queue manager, use the **endmqm** command. For example, to stop a queue manager called `saturn.queue.manager` use this command:

```
endmqm saturn.queue.manager
```

Quiesced shutdown

By default, the above command performs a *quiesced shutdown* of the specified queue manager. This may 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. The **endmqm** command will not return until the queue manager stops, which will not occur until all applications have disconnected. The **endmqm** command periodically reports *"MQSeries queue manager ending"* to the terminal while waiting for the queue manager to end.

Immediate shutdown

An *immediate shutdown* allows any current MQI calls to complete, but any new calls fail. This type of shutdown does not wait for applications to disconnect from the queue manager. Use this as the normal way to stop the queue manager, optionally after a quiesce period.

For an immediate shutdown, the command is:

```
endmqm -i saturn.queue.manager
```

Preemptive shutdown

> **Attention**
>
> 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 saturn.queue.manager
```

This stops all queue manager code immediately.

Deleting a queue manager

To delete a queue manager called `saturn.queue.manager`, first stop it, then use the following command:

```
dltmqm saturn.queue.manager
```

Note: Deleting a queue manager is a serious step, because you also delete all resources associated with that queue manager, including all queues and their messages, and all object definitions.

Working with MQSeries objects

This section describes briefly how to use MQSC commands to create, display, change, copy, and delete MQSeries objects.

You can use the MQSC facility interactively (by entering commands at the keyboard) or you can redirect the standard input device (stdin) to run a sequence of commands from a text file. The format of the commands is the same in both cases. The examples included here assume that you will be using the interactive method.

For more information about using MQSC commands, see the *MQSeries System Administration* book. For a complete description of the MQSC commands, see the *MQSeries MQSC Command Reference*.

Before you can run MQSC commands, you must have created and started the queue manager that is going to run the commands. For more information see "Creating a queue manager" on page 49.

Using the MQSC facility interactively

To start using the MQSC facility interactively, use the **runmqsc** command. Open a TACL session and enter:

```
runmqsc
```

A queue manager name has not been specified; therefore the MQSC commands will be processed by the default queue manager. Now type in any MQSC commands, as required. For example:

```
DEFINE QLOCAL (ORANGE.LOCAL.QUEUE)
```

Continuation characters must be used to indicate that a command is continued on the following line:

- A minus sign (–) indicates that the command is to be continued from the start of the following line.
- A plus sign (+) indicates that the command is to be continued from the first nonblank character on the following line.

Command input terminates with the final character of a nonblank line that is not a continuation character. You can also terminate command input explicitly by entering a semicolon (;). (This is especially useful if you accidentally enter a continuation character at the end of the final line of command input.)

Feedback from MQSC commands

When you issue commands from the MQSC facility, the queue manager returns operator messages that confirm your actions or tell you about the errors you have made. For example:

```
AMQ8006: MQSeries queue created
 .
 .
 .
AMQ8405: Syntax error detected at or near end of command segment below:-
```

The first message confirms that a queue has been created; the second indicates that you have made a syntax error.

These messages are sent to the standard output device. If you have not entered the command correctly, refer to the *MQSeries MQSC Command Reference* for the correct syntax.

Ending interactive input to MQSC

To end interactive input of MQSC commands, type `exit` or `quit` or `end` or use the EOF character `CTRL+Y`.

If you are redirecting input from other sources, such as a text file, you do not have to do this because MQSC terminates when the end of file is reached.

Creating a local queue

For an application, the local queue manager is the queue manager to which the application is connected. Queues that are managed by the local queue manager are said to be local to that queue manager.

Use the MQSC command DEFINE QLOCAL to create a definition of a local queue and also to create the data structure that is called a queue. You can also modify the queue characteristics from those of the default local queue.

In this example, the queue we define, ORANGE.LOCAL.QUEUE, is specified to have these characteristics:

- It is enabled for gets, disabled for puts, and operates on a first-in-first-out (FIFO) basis.
- It is an ‘ordinary’ queue, that is, it is not an initiation queue or a transmission queue, and it does not generate trigger messages.
- It is set so that the maximum queue depth will be 1000 messages and the maximum message length will be 2000 bytes.

The following MQSC command does this:

```
DEFINE QLOCAL (ORANGE.LOCAL.QUEUE) +
       DESCR('Queue for messages from other systems') +
       PUT (DISABLED) +
       GET (ENABLED) +
       NOTRIGGER +
       MSGDLVSQ (FIFO) +
       MAXDEPTH (1000) +
       MAXMSGL (2000) +
       USAGE (NORMAL);
```

Notes:

1. Most of these attributes are the defaults as supplied with the product. However, they are shown here for purposes of illustration. You can omit them if you are sure that the defaults are what you want or have not been changed. See also “Displaying default object attributes”.
2. USAGE (NORMAL) indicates that this queue is not an initiation queue or a transmission queue.
3. If you already have a local queue on the same queue manager with the name ORANGE.LOCAL.QUEUE, this command fails. Use the REPLACE attribute if you want to overwrite the existing definition of a queue, but see also “Changing local queue attributes” on page 57.

Displaying default object attributes

When you define an MQSeries object, it takes any attributes that you do not specify from the default object. For example, when you define a local queue, the queue inherits any attributes that you omit in the definition from the default local queue, which is called SYSTEM.DEFAULT.LOCAL.QUEUE. The default local queue is created automatically when you create the default queue manager. To see exactly what these attributes are, use the following command:

```
DISPLAY QUEUE (SYSTEM.DEFAULT.LOCAL.QUEUE)
```

Note: The syntax of this command is different from that of the corresponding **DEFINE** command.

You can selectively display attributes by specifying them individually. For example:

```
DISPLAY QUEUE (ORANGE.LOCAL.QUEUE) +
        MAXDEPTH +
        MAXMSGL +
        CURDEPTH;
```

This command displays the three specified attributes as follows:

```
AMQ8409: Display Queue details.
    QUEUE(ORANGE.LOCAL.QUEUE)
    MAXDEPTH(1000)
    MAXMSGL(2000)
    CURDEPTH(0)
```

CURDEPTH is the current queue depth; that is, the number of messages on the queue. This is a useful attribute to display, because by monitoring the queue depth you can ensure that the queue does not become full.

Copying a local queue definition

You can copy a queue definition using the LIKE attribute on the **DEFINE** command.

For example:

```
DEFINE QLOCAL (MAGENTA.QUEUE) +
       LIKE (ORANGE.LOCAL.QUEUE)
```

This command creates a queue with the same attributes as our original queue ORANGE.LOCAL.QUEUE, rather than those of the system default local queue.

You can also use this form of the **DEFINE** command to copy a queue definition, but substituting one or more changes to the attributes of the original. For example:

```
DEFINE QLOCAL (THIRD.QUEUE) +
       LIKE (ORANGE.LOCAL.QUEUE) +
       MAXMSGL(1024);
```

This command copies the attributes of the queue ORANGE.LOCAL.QUEUE to the queue THIRD.QUEUE, but specifies that the maximum message length on the new queue is to be 1024 bytes, rather than 2000.

Notes:

1. When you use the LIKE attribute on a **DEFINE** command, you are copying the queue attributes only. You are not copying the messages on the queue.
2. If you define a local queue, without specifying LIKE, it is the same as:

```
DEFINE LIKE (SYSTEM.DEFAULT.LOCAL.QUEUE)
```

Changing local queue attributes

You can change queue attributes in two ways, using either the **ALTER QLOCAL** command or the **DEFINE QLOCAL** command with the REPLACE attribute. In “Creating a local queue” on page 53, we defined the queue ORANGE.LOCAL.QUEUE. Suppose, for example, you wanted to increase the maximum message length on this queue to 10 000 bytes.

- Using the **ALTER** command:

  ```
     ALTER QLOCAL (ORANGE.LOCAL.QUEUE) MAXMSGL(10000)
  ```

  This command changes a single attribute, that of the maximum message length; all the other attributes remain the same.
- Using the **DEFINE** command with the REPLACE option, for example:

  ```
     DEFINE QLOCAL (ORANGE.LOCAL.QUEUE) MAXMSGL(10000) REPLACE
  ```

  This command changes not only the maximum message length, but all the other attributes, which are given their default values. The queue is now put enabled whereas previously it was put inhibited. Put enabled is the default, as specified by the queue SYSTEM.DEFAULT.LOCAL.QUEUE, unless you have changed it.

  If you decrease the maximum message length on an existing queue, existing messages are not affected. Any new messages, however, must meet the new criteria.

Deleting a local queue

Use the MQSC command **DELETE QLOCAL** to delete a local queue. A queue cannot be deleted if it has uncommitted messages on it. However, if the queue has one or more committed messages, and no uncommitted messages, it can be deleted only if you specify the PURGE option. For example:

```
DELETE QLOCAL (PINK.QUEUE) PURGE
```

Specifying NOPURGE instead of PURGE ensures that the queue is not deleted if it contains any committed messages.

Clearing a local queue

To delete all the messages from a local queue called MAGENTA.QUEUE, use the following command:

```
CLEAR QLOCAL (MAGENTA.QUEUE)
```

You cannot clear a queue if:

- There are uncommitted messages that have been put on the queue under syncpoint
- An application currently has the queue open.

Browsing queues

MQSeries for Compaq NSK provides a sample queue browser to enable you to look at the contents of the messages on a queue. The browser is supplied both as source and as a module that can be run. By default, the file names and paths are:

Source	$SYSTEM.ZMQSSMPL.AMQSBCG0
Executable	$SYSTEM.ZMQSSMPL.AMQSBCG

The sample takes two parameters:

Queue name	For example, SYSTEM.ADMIN.RESPQ.tpp01.
Queue manager name	For example, snooker.

For example:

```
AMQSBCG SYSTEM.ADMIN.RESPQ.tpp01 snooker
```

There are no defaults; both parameters are required. Typical results from this command are:

```
AMQSBCG - starts here
**********************

 MQCONN to snooker
 MQOPEN - 'SYSTEM.ADMIN.RESPQ.tpp01'


 MQGET of message number 1
****Message descriptor****

  StrucId  : 'MD  '  Version : 1
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 273  CodedCharSetId : 850
  Format : 'AMQMRESP'
  Priority : 5  Persistence : 1
  MsgId : X'414D5120736E6F6F6B657220202020202ED47690071A6D00'
  CorrelId : X'000000000000000000000000000000000000000000000000'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'snooker                                         '
  ** Identity Context
  UserIdentifier : 'tpp01       '
  AccountingToken :
   X'04373037300000000000000000000000000000000000000000000000000000000'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : '                            '
  PutDate  : '19941124'    PutTime  : '11184015'
  ApplOriginData : '    '

****   Message      ****

 length - 268 bytes

00000000:  736E 6F6F 6B65 7220 2020 2020 2020 2020 'snooker         '
00000010:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000020:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000030:  534E 4F4F 4B45 522E 5749 4748 542E 5443 'SNOOKER.WIGHT.TC'
00000040:  5020 2020 2020 2020 2020 2020 2020 2020 'P               '
00000050:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000060:  0000 0001 0000 0024 0000 0001 0000 0015 '.......$........'
00000070:  0000 0001 0000 0001 0000 0000 0000 0000 '................'
00000080:  0000 0003 0000 0004 0000 0028 0000 0DAD '...........(....'
00000090:  0000 0000 0000 0014 534E 4F4F 4B45 522E '........SNOOKER.'
000000A0:  5749 4748 542E 5443 5020 2020 0000 0003 'WIGHT.TCP   ....'
000000B0:  0000 0010 0000 05E7 0000 0001 0000 0004 '................'
000000C0:  0000 0050 0000 0DAE 0000 0000 0000 0039 '...P...........9'
000000D0:  2066 726F 6D20 736E 6F6F 6B65 7220 746F ' from snooker to'
000000E0:  2077 6967 6874 2076 6961 2074 6370 2F69 ' wight via tcp/i'
000000F0:  7020 2020 2020 2020 2020 2020 2020 2020 'p               '
00000100:  2020 2020 2020 2020 2000 0000           '         ...    '


 MQGET of message number 2
****Message descriptor****

  StrucId  : 'MD  '  Version : 1
  Report   : 0  MsgType : 2
  Expiry   : -1  Feedback : 0
  Encoding : 273  CodedCharSetId : 850
  Format : 'MQADMIN '
  Priority : 8  Persistence : 1
  MsgId : X'414D5120736E6F6F6B657220202020202ED476901524D200'
  CorrelId : X'414D5120736E6F6F6B657220202020202ED47690071A6D00'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'snooker                                         '
  ** Identity Context
  UserIdentifier : 'tpp01       '
  AccountingToken :
   X'04373037300000000000000000000000000000000000000000000000000000000'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : '                            '
  PutDate  : '19941124'    PutTime  : '11184035'
  ApplOriginData : '    '

****   Message      ****

 length - 36 bytes

00000000:  0000 0002 0000 0024 0000 0001 0000 0015 '.......$........'
00000010:  0000 0001 0000 0001 0000 0000 0000 0000 '................'
00000020:  0000 0000                               '....            '


 MQGET of message number 3
****Message descriptor****

  StrucId  : 'MD  '  Version : 1
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 273  CodedCharSetId : 850
  Format : 'AMQMRESP'
  Priority : 5  Persistence : 1
  MsgId : X'414D5120736E6F6F6B657220202020202ED477D62A9EA100'
  CorrelId : X'000000000000000000000000000000000000000000000000'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'snooker                                         '
  ** Identity Context
  UserIdentifier : 'trevor      '
  AccountingToken :
   X'0437303730000000000000000000000000000000000000000000000000000000'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : '                            '
  PutDate  : '19941124'    PutTime  : '11240678'
  ApplOriginData : '    '

****   Message      ****

 length - 188 bytes

00000000:  736E 6F6F 6B65 7220 2020 2020 2020 2020 'snooker         '
00000010:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000020:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000030:  534E 4F4F 4B45 522E 5749 4748 542E 5443 'SNOOKER.WIGHT.TC'
00000040:  5020 2020 2020 2020 2020 2020 2020 2020 'P               '
00000050:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000060:  0000 0001 0000 0024 0000 0001 0000 0015 '.......$........'
00000070:  0000 0001 0000 0001 0000 0000 0000 0000 '................'
00000080:  0000 0002 0000 0004 0000 0028 0000 0DAD '...........(....'
00000090:  0000 0000 0000 0014 534E 4F4F 4B45 522E '........SNOOKER.'
000000A0:  5749 4748 542E 5443 5020 2020 0000 0003 'WIGHT.TCP   ....'
000000B0:  0000 0010 0000 05E7 0000 0001           '............    '
```

```
 MQGET of message number 4
****Message descriptor****

  StrucId  : 'MD  '  Version : 1
  Report   : 0  MsgType : 2
  Expiry   : -1  Feedback : 0
  Encoding : 273  CodedCharSetId : 850
  Format : 'MQADMIN '
  Priority : 8  Persistence : 1
  MsgId : X'414D5120736E6F6F6B657220202020202ED477D63826C000'
  CorrelId : X'414D5120736E6F6F6B657220202020202ED477D62A9EA100'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'snooker                                         '
  ** Identity Context
  UserIdentifier : 'tiger       '
  AccountingToken :
   X'0437303730000000000000000000000000000000000000000000000000000000'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : '                            '
  PutDate  : '19941124'    PutTime  : '11240694'
  ApplOriginData : '    '

****   Message      ****

 length - 36 bytes

00000000:  0000 0002 0000 0024 0000 0001 0000 0015 '.......$........'
00000010:  0000 0001 0000 0001 0000 0000 0000 0000 '................'
00000020:  0000 0000                               '....            '



 No more messages
 MQCLOSE
 MQDISC
```

Chapter 9. Obtaining additional information

This chapter describes the documentation for MQSeries for Compaq NSK. It starts with a list of the publications, and then discusses:

- "Publications supplied with the product"
- "HTML and PDF books on the World Wide Web" on page 65
- "BookManager® CD-ROMs" on page 66

MQSeries for Compaq NSK is described in the following books:

Table 2. MQSeries for Compaq NSK books

Order Number	Title
	MQSeries for Compaq NSK Specific Books
GC34-5887	*MQSeries for Compaq NSK V5.1 Quick Beginnings*
SC34-5886	*MQSeries for Compaq NSK V5.1 System Administration Guide*
	MQSeries Family Books
SC33-1872	*MQSeries Intercommunication*
SC34-5349	*MQSeries Queue Manager Clusters*
GC33-1632	*MQSeries Clients*
SC33-1873	*MQSeries System Administration*
SC33-1369	*MQSeries MQSC Command Reference*
SC33-1482	*MQSeries Programmable System Management*
SC34-5390	*MQSeries Administration Interface Programming Guide and Reference*
GC33-1876	*MQSeries Messages*
SC33-0807	*MQSeries Application Programming Guide*
SC33-1673	*MQSeries Application Programming Reference*
SX33-6095	*MQSeries Programming Interfaces Reference Summary*
SC33-1877	*MQSeries Using C++*

Publications supplied with the product

The MQSeries for Compaq NSK V5.1 product includes the following:

- Hardcopy books. See "Hardcopy books" on page 64.
- PDF files. See "PDF files" on page 64.

Additional information

The HTML versions of the MQSeries for Compaq NSK V5.1 product books and many other MQSeries books are available on the World Wide Web. See "HTML and PDF books on the World Wide Web" on page 65 for more information.

Hardcopy books

The book that you are reading now is *MQSeries for Compaq NSK, V5.1 Quick Beginnings*. This book and the *MQSeries for Compaq NSK V5.1 System Administration Guide* are the only ones that are supplied in hardcopy with the product. However, all books listed in Table 2 on page 63 are available for you to order or print.

You can order publications from the IBMLink™ Web site at:

```
http://www.ibm.com/ibmlink
```

In the United States, you can also order publications by dialing **1-800-879-2755**.

In Canada, you can order publications by dialing **1-800-IBM-4YOU (1-800-426-4968).**

For further information about ordering publications contact your IBM authorized dealer or marketing representative.

For information about printing books, see "PDF files".

PDF files

A PDF (Portable Document Format), corresponding to each hardcopy book (*MQSeries for Compaq NSK V5.1 Quick Beginnings* and the *MQSeries for Compaq NSK V5.1 System Administration Guide*) is supplied with the product. These PDF files are installed in the ZMQSSYS subvolume in the language selected during the installation. On the tape or archive file, these books are stored in subvolumes called Z51S*lang* where *lang* is the four letter acronym for the language. (For example, Z51SENUS contains the documents for U.S. English). To access or print the PDFs, you will need to transfer the files to a PC with an Adobe Acrobat Reader. For more information, see "Restoring the README file and user documentation" on page 15.

The PDFs are available in U.S. English and also in some or all of the following national languages:
- ENUS (English)
- PTBR (Brazilian Portuguese)
- FRFR (French)
- DEDE (German)
- ITIT (Italian)

- JAJP (Japanese)
- KOKR (Korean)
- ESES (Spanish)
- ZHCN (Simplified Chinese)

HTML and PDF books on the World Wide Web

The MQSeries books are available on the World Wide Web in PDF and HTML format. The MQSeries product family Web site is at:

```
http://www.ibm.com/software/mqseries/
```

By following links from this Web site you can:

- Obtain latest information about the MQSeries product family.
- Access the MQSeries books in HTML and PDF formats.
- Download MQSeries SupportPacs™.

PDF files on the World Wide Web

Table 3 shows the file names used for PDF files for the MQSeries family books.

Table 3. MQSeries publications – file names

Book	File Name
MQSeries V5.2 Release Guide	AMQZAY00
MQSeries Intercommunication	CSQZAE04
MQSeries Queue Manager Clusters	CSQZAH02
MQSeries Clients	CSQZAF04
MQSeries System Administration	AMQZAG01
MQSeries MQSC Command Reference	CSQZAJ04
MQSeries Programmable System Management	CSQZAI03
MQSeries Administration Interface Programming Guide and Reference	CSQZAT01
MQSeries Messages	AMQZAO01
MQSeries Application Programming Guide	CSQZAL04
MQSeries Application Programming Reference	CSQZAK04
MQSeries Programming Interfaces Reference Summary	CSQZAM04
MQSeries Using C++	AMQZAN03

Hardcopy Books

HTML books available on the World Wide Web

You can view the MQSeries documentation (including *MQSeries for Compaq NSK V5.1 Quick Beginnings* and the *MQSeries for Compaq NSK V5.1 System Administration Guide*) in HTML format directly from the World Wide Web. All books except for the *MQSeries Programming Interfaces Reference Summary* are available in U.S. English and also in some or all of the following national languages:

- French
- Brazilian Portuguese
- German
- Italian
- Japanese
- Korean
- Spanish
- Simplified Chinese

When you read the books in HTML, you can follow hypertext links from one book to another. If you are reading translated books and link to a book that is not available in your national language, the U.S. English version of the book will be opened instead.

BookManager® CD-ROMs

The MQSeries library is supplied in IBM BookManager® format on a variety of online library collection kits, including the *Transaction Processing and Data* collection kit, SK2T-0730. You can view the softcopy books in IBM BookManager format using the following IBM licensed programs:

BookManager READ/2
BookManager READ/6000
BookManager READ/DOS
BookManager READ/MVS
BookManager READ/VM
BookManager READ for Windows®

Related publications

- *SNAX/APC Planning and Configuration Manual*, (Compaq Part No. 098289)

  SNAX/APC provides LU 6.2 support for the Compaq implementation of SNA. This guide explains how to install and configure SNAX/APC.
- *SCF Reference Manual for SNAX/APC*, (Compaq Part No. 064525)

  SNAX/APC provides LU 6.2 support for the Compaq implementation of SNA. This guide explains the Subsystem Control Facility (SCF) interactive interface that lets operators and network managers configure and control SNAX/APC.
- *Pathway System Management Guide*, (Compaq Part No. 096881)

This guide presents guidelines for configuring and controlling Pathway transaction processing systems.

- *Introduction to NonStop Transaction Manager/MP (TM/MP)*, (Compaq Part No. 085812)

  This guide describes how to use the TMF subsystem to protect your business transactions and the integrity of your databases.
- *Introduction to Compaq Networking and Data Communications*, (Compaq Part No. 093148)

  This guide provides an overview of Compaq networking and data communications concepts, tasks, products, and manuals.
- *Intersystem Communications Environment (ICE) Installation Guide*, (Version 3 Release 2 or later edition)

  This guide describes how to install ICE and configure the ICE start-up parameters. (ICE provides LU 6.2 support for Insessions's implementation of SNA.)
- *Intersystem Communications Environment (ICE) Administrator's Guide*, (Version 3 Release 2, or later edition)

  This guide describes how to configure and operate ICE, its interfaces, and its utilities.

Part 3. Appendixes

Appendix A. MQSeries for Compaq NSK at a glance

Program and part number

- 5724-A39 MQSeries for Compaq NSK, Version 5 Release 1, part number 0791003

Hardware requirements

Minimum hardware requirements are:

- Any of the Compaq NSK range of machines supported by Guardian D45 or later D4x, G06 or later G0x.
- Specific hardware in support of user-selected network transport protocols.

You are also recommended to have one or more mirrored data disks with specified space requirements for TMF audit space and the MQSeries database.

Software requirements

Minimum software requirements are:

- Compaq NSK Guardian D45 or later D4x (K-series hardware) and G06 or later G0x (S-series hardware) operating systems, including TM/MP (TMF), ENSCRIBE, and EMS.
- TS/MP (PATHWAY) to match operating system.
- SCF for configuration, command, and control of TCP and SNA network transports.

For SNA connectivity:

- SNAX/APC and SNAX/XF or SNAX/APN to match operating system

or

- Insession ICE Version 3.2 or later

For TCP/IP connectivity:

- TCP/IP to match operating system.

To use the OSS-based parts of MQSeries (MQI bindings, OSS applications, Java bindings, you require the OSS product version compatible with the operating system.

Software requirements

To use Java bindings, you need a minimum of NonStop Server for Java 1.5. Earlier versions will not work.

Transaction logging is maintained with the Compaq TM/MP (TMF) product.

Security

MQSeries for Compaq NSK uses the security features of the NSK file system, which provide file-level access control to USER and GROUP for read, write, execute, and purge operations. SAFEGUARD is not required for the use of MQSeries for Compaq NSK; however, the product is compatible with a SAFEGUARD environment.

All MQSeries resources are owned by a single user ID in group MQM. To administer MQSeries with either the SCOBOL menus or **runmqsc**, you must be logged in with a user ID assigned or linked to the MQM group.

Maintenance functions

MQSeries functions with:

- The Message Queue Management (MQM) facility using SCOBOL requester configuration screens in a PATHWAY environment.
- The **runmqsc** command-line interface.
- SCF utility for configuration, command and control functionality to maintain TCP/IP and SNA environments for Compaq network protocol offerings.
- ICE utilities provided with that product for control of ICE LU 6.2 interface.
- MQSeries Explorer (not included with MQSeries for Compaq NSK).
- Any other product or utility that uses standard PCF commands for remote administration.

Compatibility

The MQI for MQSeries for Compaq NSK V5.1, is compatible with existing applications running on MQSeries for Tandem NonStop Kernel V2.2.0.1, with maintenance fix PTF U473441.

Supported compilers

MQSeries for Compaq NSK V5.1 supports the following compilers:

- C (native and non-native)
- C++ (native only)
- COBOL-85 (native and non-native)

- TAL (non-native)
- NonStop Java Version 1.5 or later

C, C++, COBOL-85 and TAL must be compatible with the base operating system version.

MQSeries for Compaq NSK V5.1 is built using the Common Runtime Environment (CRE) to link all objects. This method imposes the following requirements on users of versions of the MQI prior to Version 2.2.0.1:

1. All pre-D45 COBOL and C object code must be recompiled with the D45 (or later) compiler to integrate the CRE linkage.
2. All pre-D45 TAL object code must be recompiled with a D45 (or later) compiler and you must ensure that the TAL program is compliant with the special programming considerations specified in the *Common Run-time Environment Programmer's Guide*. More detailed information on each of these programming considerations is provided in the *TAL Programmer's Guide*.
3. For object code produced with native compilers on D45, a separate binding is provided.
4. C programs must use the WIDE memory model (32–bit integers).
5. COBOL programs must conform to the requirements of the CRE.
6. In TAL programs, all integers passed to the MQI functions must be 32 bits (or be cast to 32 bit with the $INT32() macro).

The MQSeries programs themselves are compiled and linked using the native mode tools for Guardian NSK. Native mode applications normally link with the queue manager SRL directly unless the application already uses a Private SRL. In this case, since applications are restricted to using at most a single Private SRL, the application must either link with the static MQI binding library, or the code that resides in the application's private SRL must be combined with the MQSeries SRL into a new Private SRL.

License management

You must enter the system type to define the program entitlement. This parameter can be entered at installation time or at any subsequent time in the event of a license upgrade being purchased. At startup this value is checked against the physical Compaq machine configuration. If the license registration and program entitlement are insufficient, a warning message is issued.

Language selection

A supplied message text file is encoded in the 7-bit character set that is native to the Compaq NSK operating system. MQSeries for Compaq NSK lets the national language be specified when the product is installed. The message language defaults to U.S. English. Multiple installations with multiple languages are possible.

Internationalization

MQSeries for Compaq NSK lets the CCSID be specified when the queue manager is created (Although the CCSID can also be changed after the queue manager is created.) The queue manager CCSID defaults to 819. MQSeries for Compaq NSK supports character-set conversion into the configured CCSID of the queue manager. For information about the CCSIDs that can be specified for an MQSeries for Compaq NSK queue manager, including those that provide support for the euro character, see the *MQSeries Application Programming Reference*.

Appendix B. The MQSeries control commands

This appendix contains reference material for the control commands used while installing, migrating to or verifying MQSeries for Compaq NSK. For descriptions of each control command used with MQSeries for Compaq NSK, see the *MQSeries for Compaq NSK V5.1 System Administration Guide*.

Control commands summary

The following control commands are supported by MQSeries for Compaq NSK via TACL macros and compiled programs:

- crtmqm (create queue manager)
- dltmqm (delete queue manager)
- endmqm (end queue manager)
- instmqm (install MQSeries for Compaq NSK)
- runmqsc (run MQSeries commands)
- strmqm (start queue manager)
- upgmqm (upgrade V2.2.0.1 queue manager)

Detailed descriptions of these commands are provided in the remainder of this appendix.

Notes:

1. Flags, which are single-character identifiers preceded by a dash (for example, -v on the **runmqsc** command), must be specified in lower case.
2. Usage messages are displayed if control commands are invoked with -?, ?, or with no parameters when parameters are expected.

Using names

The names for the following MQSeries objects can be a maximum of 48 characters:

- Queue managers
- Queues
- Process definitions

The maximum length of channel names is 20 characters.

The characters that can be used for all MQSeries names are:

- Upper case A - Z
- Lower case a - z
- Numerics 0 - 9
- Period (.)

- Underscore (_)
- Forward slash (/)
- Percent sign (%)

Notes:

1. Forward slash and percent are special characters. If you use either of these characters in a name, the name must be enclosed in double quotation marks whenever it is used.
2. Leading or embedded blanks are not allowed.
3. National language characters are not allowed.
4. Names may be enclosed in double quotation marks, but this is essential only if special characters are included in the name.

crtmqm (Create queue manager)

Purpose

Use the **crtmqm** command to create a local queue manager. Once a queue manager has been created, use the **strmqm** command to start it.

Creating a queue manager automatically creates the associated system and default objects.

Syntax

```
►►──crtmqm──┬──────────┬──┬──────────────────────────────┬──┬─────────────┬──────────────►
            └─ -c Text─┘  └─ -d DefaultTransmissionQueue─┘  └─ -e NumECs──┘

►──┬──────────────────────────┬──┬──────────┬──┬────────────────┬──────────────────────────►
   └─ -h MaximumHandleLimit───┘  └─ -l CCSID─┘  └─ -m MachIniFile─┘

►──┬───────────────────┬──┬──────┬──┬───────────────────┬──────────────────────────────────►
   └─ -p DefaultPrefix─┘  └─ -q ─┘  └─ -t IntervalValue─┘

►──┬─────────────────────┬──┬──────────────────────────────────┬──┬──────┬─────────────────►
   └─ -u DeadLetterQueue─┘  └─ -x MaximumUncommittedMessages───┘  └─ -z ─┘

►─ -n PATHMONProcessName── -o HomeTerminalName── -s StatusServerName──────────────────────►

►─ -v QueueServerName──QMgrName──────────────────────────────────────────────────────────►◄
```

Required parameters

-n *PATHMONProcessName*
: The process name of the TS/MP PATHMON process for the queue manager. This process name must be unique in the system.

-o *HomeTerminalName*
: Home terminal device name. ($DDDD.#SS). For example, $TRM1.#A.

-s *StatusServerName*
: The process name to be given to the default status server for the queue manager. The process name must be unique in the system.

-v *QueueServerName*
: Specifies a unique process name to be given to the default queue server process for this queue manager.

QMgrName
The name of the queue manager to be created. The name can contain up to 48 characters. This must be the last item in the command.

Optional parameters

-c *Text*
Some text (up to 64 characters) that describes this queue manager. The default is all blanks.

If special characters are required, the description must be enclosed in double quotation marks.

-d *DefaultTransmissionQueue*
The name of the local transmission queue that remote messages are placed on if a transmission queue is not explicitly defined for their destination. There is no default.

-e *NumECs*
The number of EC processes in the queue manager. The default is 1.

-h *MaximumHandleLimit*
In MQSeries for Compaq NSK, this parameter is ignored.

The maximum number of handles that any one application can have open at the same time. Specify a value in the range 1 through 999 999 999. The default value is 256.

-l *CCSID*
Qmgr CCSID. The default value is 819.

-m *MachIniFile*
Overrides the default MQSINI file location and that specified in the environment variable MQMACHINIFILE.

-p *DefaultPrefix*
The volume for the queue manager. Overrides the `QMDefaultVolume` entry in the MQSINI file.

-q Specifies that this queue manager is to be made the default queue manager. The new queue manager replaces any existing queue manager as the default.

If you accidentally use this flag and wish to revert to an existing queue manager as the default queue manager, you can edit the `DefaultQueueManager` stanza in the MQSeries configuration file.

-t *IntervalValue*
The trigger-time interval in milliseconds for all queues controlled by this queue manager. This value specifies the time after the receipt of a trigger-generating message when triggering is suspended. That is, if the arrival of a message on a queue causes a trigger message to be put on the

initiation queue, any message arriving on the same queue within the specified interval does not generate another trigger message.

You can use the trigger time interval to ensure that your application is allowed sufficient time to deal with a trigger condition before it is alerted to deal with another on the same queue. You may wish to see all trigger events that happen; if so, set a low or zero value in this field.

Specify a value in the range 0 through 999 999 999. The default is 999 999 999 milliseconds, a time of more than 11 days. Allowing the default to be taken effectively means that triggering is disabled after the first trigger message. However, triggering can be reenabled by an application servicing the queue using an alter queue command to reset the trigger attribute.

-u *DeadLetterQueue*
The name of the local queue that is to be used as the dead-letter (undelivered-message) queue. Messages are put on this queue if they cannot be routed to their correct destination.

By default, there is no dead-letter queue.

-x *MaximumUncommittedMessages*
In MQSeries for Compaq NSK, this parameter is ignored.

Specifies the maximum number of uncommitted messages under any one syncpoint. That is, the sum of:
- The number of messages that can be retrieved from queues
- The number of messages that can be put on queues
- Any trigger messages generated within this unit of work

This limit does not apply to messages that are retrieved or put outside syncpoint control.

Specify a value in the range 1 through 10 000. The default value is 1000 uncommitted messages.

-z Suppresses error messages.

This flag is normally used within MQSeries to suppress unwanted error messages. As use of this flag could result in loss of information, you are recommended not to use it when entering commands on a command line.

Return codes

0	Queue manager created
8	Queue manager already exists
49	Queue manager stopping
69	Storage not available
70	Queue space not available
71	Unexpected error
72	Queue manager name error

111 Queue manager created. However, there was a problem processing the default queue manager definition in the product configuration file. The default queue manager specification may be incorrect.

Examples

1. This command creates a default queue manager named `Paint.queue.manager`, which is given a description of `Paint Shop`:

```
crtmqm -c "Paint Shop" -n $PANT -o $TRM1.#A -s $PNT1 -v $PQS1 Paint.queue.manager
```

2. In this example, another queue manager, `travel`, is created. The trigger interval is defined as 5000 milliseconds (or 5 seconds) and its dead-letter queue is specified as SYSTEM.DEAD.LETTER.QUEUE.

```
crtmqm -t 5000 -u SYSTEM.DEAD.LETTER.QUEUE -n $TRAV -o $TRM1.#A -s $TRV1 -v $TQS1 travel
```

Once a trigger event is generated, further trigger events are disabled for five seconds.

Related commands

strmqm	Start queue manager
endmqm	End queue manager
dltmqm	Delete queue manager

dltmqm (Delete queue manager)

Purpose

Use the **dltmqm** command to delete a specified queue manager. All objects associated with this queue manager are also deleted. Before you can delete a queue manager you must end it using the **endmqm** command.

Syntax

```
►►──dltmqm──┬──────┬──QMgrName──────────────────────────────────────►◄
            └─ -z ─┘
```

Required parameters

QMgrName
Specifies the name of the queue manager to be deleted.

Optional parameters

-z Suppresses error messages.

Return codes

0	Queue manager deleted
5	Queue manager running
16	Queue manager does not exist
69	Storage not available
71	Unexpected error
72	Queue manager name error
112	Queue manager deleted. However, there was a problem processing the default queue manager definition in the product configuration file. The default queue manager specification may be incorrect.

Examples

1. The following command deletes the queue manager `saturn.queue.manager`:

```
dltmqm saturn.queue.manager
```

2. The following command deletes the queue manager `travel` and also suppresses any messages caused by the command:

dltmqm

```
dltmqm -z travel
```

Related commands

crtmqm	Create queue manager
strmqm	Start queue manager
endmqm	End queue manager

endmqm (End queue manager)

Purpose

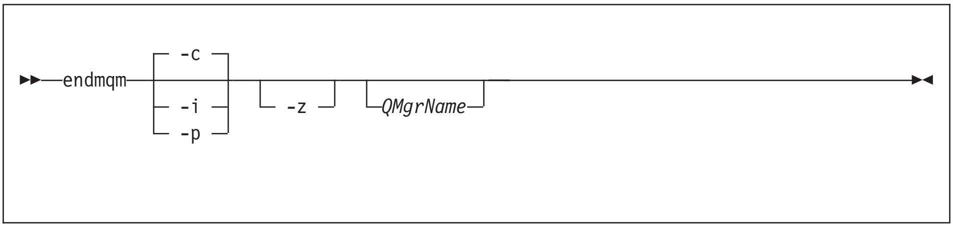
Use the **endmqm** command to end (stop) a specified local queue manager. This command stops a queue manager in one of three modes:

- Normal or quiesced shutdown
- Immediate shutdown
- Preemptive shutdown

The attributes of the queue manager and the objects associated with it are not affected. You can restart the queue manager using the **strmqm** (Start queue manager) command.

To delete a queue manager, you must stop it and then use the **dltmqm** (Delete queue manager) command.

Syntax



Optional parameters

QMgrName
Is the name of the message queue manager to be stopped. If no name is specified, the default queue manager is stopped.

-c Controlled (or quiesced) shutdown. The queue manager stops but only after all applications have disconnected. Any MQI calls currently being processed are completed. This is the default.

-i Immediate shutdown. The queue manager stops after it has completed all the MQI calls currently being processed. Any MQI requests issued after the command has been issued fail. Any incomplete units of work are rolled back immediately.

-p Preemptive shutdown.

Use this type of shutdown only in exceptional circumstances. For example, when a queue manager does not stop as a result of a normal **endmqm** command.

The queue manager stops without waiting for applications to disconnect or for MQI calls to complete. This can give unpredictable results for MQI applications. All processes in the queue manager that fail to stop are terminated 30 seconds after the command is issued.

-z Suppresses error messages on the command.

Return codes

0	Queue manager ended
16	Queue manager does not exist
36	Invalid arguments
40	Queue manager not available
69	Storage not available
71	Unexpected error
72	Queue manager name error

Examples

The following examples show commands that end (stop) the specified queue managers.

1. This command ends the default queue manager in a controlled way. All applications currently connected are allowed to disconnect.

```
endmqm
```

2. This command ends the queue manager named `saturn.queue.manager` immediately. All current MQI calls complete, but no new ones are allowed.

```
endmqm -i saturn.queue.manager
```

Related commands

crtmqm	Create a queue manager
strmqm	Start a queue manager
dltmqm	Delete a queue manager

instmqm (Install MQSeries for Compaq NSK)

Purpose

Use the **instmqm** command to install MQSeries for Compaq NSK or update license information.

Syntax

```
►►──instmqm──┬──────┬──────────────────────────────────────────────────────►◄
             └─ -l ─┘
```

Optional parameters

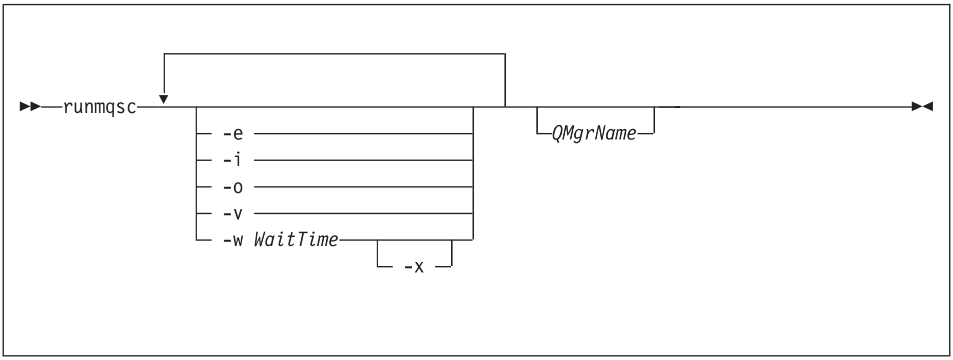
-l Invokes **instmqm** for license information updates.

runmqsc (Run MQSeries commands)

Purpose

Use the **runmqsc** command to issue MQSC commands to a queue manager. MQSC commands enable you to perform administration tasks, for example defining, altering, or deleting a local queue object. MQSC commands and their syntax are described in the *MQSeries MQSC Command Reference*.

Syntax



Description

You can invoke the **runmqsc** command in three modes:

Verify mode
MQSC commands are verified but not actually run. An output report is generated indicating the success or failure of each command. This mode is only available on a local queue manager.

Direct mode
MQSC commands are sent directly to a local queue manager.

Indirect mode
MQSC commands are run on a remote queue manager. These commands are put on the command queue on a remote queue manager and are run in the order in which they were queued. Reports from the commands are returned to the local queue manager.

The **runmqsc** command takes its input from the standard IN file. When the commands are processed, the results and a summary are put into a report that is sent to the standard OUT file.

By taking the standard IN file from the keyboard, you can enter MQSC commands interactively.

By redirecting the input from a file you can run a sequence of frequently-used commands contained in the file. You can also redirect the output report to a file.

Note: To run this command, your user ID must belong to user group MQM.

Optional parameters

-e Prevents source text for the MQSC commands from being copied into a report. This is useful when you enter commands interactively.

–i Input file name

–o Output file name

-v Specifies verification mode; this verifies the specified commands without performing the actions. This mode is available locally only. The -w and -x flags are ignored if they are specified at the same time.

-w *WaitTime*

Specifies indirect mode, that is, the MQSC commands are to be run on another queue manager. You must have the required channel and transmission queues set up for this.

WaitTime

Specifies the time, in seconds, that **runmqsc** waits for replies. Any replies received after this are discarded, however, the MQSC commands are still run. Specify a time between 1 and 999 999 seconds.

Each command is sent as an Escape PCF to the command queue (SYSTEM.ADMIN.COMMAND.QUEUE) of the target queue manager.

The replies are received on queue SYSTEM.MQSC.REPLY.QUEUE and the outcome is added to the report. This can be defined as either a local queue or a model queue.

Indirect mode operation is performed through the default queue manager.

This flag is ignored if the -v flag is specified.

-x Specifies that the target queue manager is running under . This flag applies only in indirect mode. The -w flag must also be specified. In indirect mode, the MQSC commands are written in a form suitable for the MQSeries for MVS/ESA command queue.

QMgrName

Specifies the name of the target queue manager on which the MQSC commands are to be run. If omitted, the MQSC commands run on the default queue manager.

Return codes

00 MQSC command file processed successfully.

10 MQSC command file processed with errors-report contains reasons for failing commands.

20 Error-MQSC command file not run.

Examples

1. Enter this command at the TACL prompt:

```
runmqsc
```

   Now you can enter MQSC commands directly. No queue manager name was specified, therefore the MQSC commands are processed on the default queue manager.

2. The following example shows how to specify that MQSC commands are verified only:

```
runmqsc -i $SYSTEM.CONFIG.MQSCIN  -v BANK
```

   This verifies the MQSC command file $SYSTEM.CONFIG.MQSCIN. The queue manager name is `BANK`. The output is displayed in the current window.

3. This command runs an MQSC command file against the queue manager called `BANK`:

```
runmqsc -i MQSCFILE -o $TEST.MQ.MQSCOUT BANK
```

   In this example, the output is directed to file `$TEST.MQ.MQSCOUT`. The input file is `MQSCFILE` in the current subvolume.

strmqm (Start queue manager)

Purpose

Use the **strmqm** command to start a local queue manager. Only after the queue manager is available to process connections or other requests, will the **strmqm** command return to the command line.

Syntax

```
►►──strmqm──┬──────┬──┬──────┬──┬──────────┬──────────────────────────────►◄
            └─ -c ─┘  └─ -z ─┘  └─QMgrName─┘
```

Optional parameters

-c Starts the queue manager, redefines the default and system objects, then stops the queue manager. (The default and system objects for a queue manager are created initially by the crtmqm command.) Any existing system and default objects belonging to the queue manager are replaced if you specify this flag.

QMgrName
Specifies the name of a local queue manager to be started. If omitted, the default queue manager is started.

-z Suppresses error messages.

This flag is used within MQSeries to suppress unwanted error messages. Because using this flag could result in loss of information, you should not use it when entering commands on a command line.

Return codes

0	Queue manager started
3	Queue manager being created
5	Queue manager running
16	Queue manager does not exist
49	Queue manager stopping
69	Storage not available
71	Unexpected error
72	Queue manager name error

Examples

The following command starts the queue manager `account`:

```
strmqm account
```

Related commands

crtmqm	Create a queue manager
dltmqm	Delete a queue manager
endmqm	End a queue manager

upgmqm (Upgrade V2.2.0.1 queue manager)

Purpose

This command upgrades a Version 2.2.0.1 queue manager for use with MQSeries for Compaq NSK V5.1. The utility invoked by **upgmqm** sends progress messages to the terminal from which it is invoked. When the utility completes, the named queue manager is ready for use with MQSeries for Compaq NSK V5.1. Queue manager attributes new in Version 5.1 are set to their default values. You can alter these in the usual way.

Because the functionality of the new Version 5.1 status server is different from the MQSS Server in Version 2.2.0.1, this upgrade deletes all existing MQS-Status*nn* server classes from your existing PATHWAY configuration and replaces them with one default MQS-Status00 server class and one default MQS-Queue00 server class. This means that after you have upgraded your queue manager, any objects that are assigned to your present MQSS server processes will need to be re-assigned to your new status server processes or queue server processes, depending on which is appropriate. (In Version 5.1, only local queues are assigned to the queue servers and all other objects are assigned to the status servers.) You may also need to change any existing scripts that refer to your status server classes in PATHWAY.

If you elected not to clean up during the upgrade, you can delete the following files at your convenience:

Table 4. Examples of files that can be deleted after an upgrade

Location	Files	Example
Subvolume indicated by -p option on **upgmqm**	All files	$VOL.scratch
Queue managers 'S' subvolume	Z*	$VOL.myv2201S

These examples assume that your existing queue manager resides on $VOL.myv2201? subvolumes.

Note: If a Version 2.2.0.1 queue manager is not upgraded using **upgmqm**, no control commands will work for that queue manager. This includes **dltmqm**, which will fail with trying to open the principal database. A queue manager from Version 2.2.0.1 no longer needed under Version 5.1 must be removed prior to the code upgrade, or upgraded using **upgmqm**, then removed.

Syntax

```
►►──upgmqm──-m QMgrName──-v DefaultQueueServer──-p SubvolumePath──────────────────►

►──-s DefaultStatusServer─────────────────────────────────────────────────────────►◄
```

Required parameters

-m *QMgrName*
Specifies the name of the queue manager to which the **upgmqm** utility is to be applied.

-v *DefaultQueueServer*
A unique process name for the default queue server for the queue manager.

-p *SubvolumePath*
A subvolume path ($VOL.SUBVOL) that the upgrade utility can use for working files. This subvolume **must** be on the same volume as the queue manager. Specify only the subvolume part of the path; do not specify the volume name. The **upgmqm** command checks that the subvolume does not already exist, before accepting the subvolume as valid.

-s *DefaultStatusServer*
A unique process name for the default status server for the queue manager.

Examples

This example upgrades a Version 2.2.0.1 queue manager `Myv2201qm` to have a a default queue server name of `$MYQS` and a default status server name of `$MYSS`. The upgrade utility uses subvolume `$VOL.scratch` for the working files (where `$VOL` is the volume on which the queue manager resides):

```
upgmqm -m Myv2201qm -p scratch -s $MYSS -v $MYQS
```

Appendix C. Setting TACL environment variables for MQSeries for Compaq NSK

MQSeries creates and uses a number of Compaq NSK environment variables, or PARAMs. When setting these PARAMs, consider the following points:

- The MQDEFAULTPREFIX PARAM must be present in the environment of all programs. The TS/MP (PATHWAY) configuration established automatically by the **crtmqm** command ensures that these PARAMs are set correctly for any queue manager server processes. Users of MQSeries applications and control commands must ensure that the TACLs and TS/MP configurations used also specify these variables.
- You are recommended to include the PARAM statements in your TACLCSTM files so that, when you log on, these PARAMs are created correctly, and any programs run from the TACL inherit the correct values. The following environment variable should also be modified to allow location of MQSeries executables:

```
#SET #PMSEARCH $SYSTEM.ZMQSEXE [#PMSEARCH]
```

MQCONNECTTYPE

This PARAM, if present, can be used to disable the ability of applications to use FASTPATH connections. If this PARAM is set to the value STANDARD, applications are only able to use STANDARD connections, even if they request FASTPATH. Any other value is treated as if the PARAM was not specified (and therefore applications, if they request it, are able to use FASTPATH connections).

MQDEFAULTPREFIX

The name of the volume containing the installed subvolume, ZMQSSYS. This PARAM must be correctly defined in all environments. For example:

```
   PARAM MQDEFAULTPREFIX $data00
```

MQEMSEVENTS

This PARAM enables MQSeries EMS events. For example, to switch on all EMS events for MQSeries, you set the PARAM MQEMSEVENTS as follows:

```
PARAM MQEMSEVENTS 127
```

MQMACHINIFILE

The location of the MQSINI file for the installation. The default value is *MQDEFAULTPREFIX*.ZMQSSYS.MQSINI. This PARAM is required only if a nondefault location is required. For example:

```
PARAM MQMACHINIFILE $data00.altinst.mqsini
```

MQRDF

If this PARAM is set ON, MQSeries changes the behavior of the delete operation to work with RDF for audited files. If this PARAM is not defined, or is set to anything other than ON, the MQSeries delete operation functions normally. If used, this PARAM must be set in the TACL environment of any user that runs administrative programs, and in the environment of all TS/MP server classes configured in the queue manager.

MQRDFFUPPROCESSNAME

This PARAM is interpreted only by the **cleanrdf** utility. It is used to specify a Guardian process name that will be assigned to the FUP server process that the **cleanrdf** utility creates. If this PARAM is not defined, the FUP server process name is assigned by the operating system.

MQRDFFUPPROGNAME

This PARAM is interpreted only by the **cleanrdf** utility. It is used to specify the fully qualified name of the FUP executable file to be used by the utility. The default value is `<default system name>.$SYSTEM.SYS00.FUP`.

MQSNOAUT

If this PARAM is set to 1 when **crtmqm** is run, the new queue manager is created with the OAM disabled. For example:

```
PARAM MQSNOAUT 1
```

MQTCPIPPORTNUMBER

If this PARAM is set then the TCP/IP Listener process uses it to find out which port to listen on. If the PARAM is not specified, the port is determined from the QMINI file TCP/IP stanza.

MQTRANSACTIONLIFE
This parameter controls the maximum lifetime (in seconds) for an inactive TMF transaction. Message Channel Agents (MCAs) and repository managers and servers use this value to limit long-running TMF transactions. The default value is 10 seconds.

SAVE-ENVIRONMENT ON
Required when running application programs to ensure the Common Run-Time Environment (CRE) passes PARAMs from the environment to the application program. For example:

```
PARAM SAVE-ENVIRONMENT ON
```

If this PARAM is not set, applications receive return code 2058, indicating a queue manager name error.

This PARAM is required for TAL or COBOL applications only, running as non-native programs.

Queue server tuning parameters

The following PARAMS, if defined in the TACL environment of a queue server, can be used to override the built-in defaults of the queue server, for various housekeeping operations:

MQQSHKEEPINT
If this PARAM is set, a numeric value in seconds may be specified to override the default housekeeping interval (60s) of the queue server. The housekeeping interval controls the frequency at which the queue server looks at queues to detect expired messages, and examines its memory utilization in order to optimize operations.

MQQSSIGTIMEOUT
If this PARAM is set, a numeric value is seconds my be specified to override the default timeout (60s) for the delivery of a signal IPC to an application that has initiated an MQGET with the MQGMO_SET_SIGNAL option. If a queue server is unable to deliver the signal within this timeout (once the conditions for generating the signal have been met) the queue server logs the fact and then cancels the signal.

MQQSMAXBATCHEXPIRE
If this PARAM is set, a numeric value may be specified to override the default maximum number of expired persistent messages (100) that will be discarded within a single transaction during housekeeping by a queue server. When persistent messages expire, they must be

physically removed from the queue databases, which requires an internal TM/MP transaction. This PARAM allows control over the maximum number of message that will be removed within a single TM/MP transaction.

MQQSMAXMSGSEXPIRE

If this PARAM is set, a numeric value may be specified to override the default maximum number of expired messages (300) that will be detected and discarded within a single housekeeping instance of a queue server.

Appendix D. PAK file installation examples

This appendix includes the following examples:
- Figure 3 on page 100
- Figure 4 on page 105

For information about the availability of the Compaq UNPAK utility, see the release notes supplied with MQSeries for Compaq NSK.

An example PAK file installation

To install from a PAK file:

1. Unpack the installation files from $*.install subvolume on the tape as follows:

```
$AUDIT1 ZMQSSYS 70> unpak $vol.subvol.archive,$*.install.*,map names
  $*.*.* to $audit1.zmqssys.*, nounload, listall, myid
```

   where *$vol.subvol.archive* is the PAK file and `$audit1.zmqssys` is the subvolume from where you want to run the INSTMQM program.

2. Run the INSTMQM program as shown in Figure 3 on page 100.

PAK file installation

```
$AUDIT1 ZMQSSYS 71> instmqm


IBM MQSeries for Compaq NSK, Version 5.1
Installation and License update program.


@(#) Licensed Materials - Property of IBM  5724-A39
(C) Copyright IBM Corp. 1993, 2001 All Rights Reserved
US Government Users Restricted Rights - Use, duplication or
disclosure restricted by GSA ADP Schedule Contract with IBM Corp.


Product installation selected...
You may stop the installation by entering
"quit" at any prompt.
Where there is a default for a parameter, you may
select it by pressing the Enter key on its own.

Phase 1: Collection of license information
```

Figure 3. An example PAK file installation (Part 1 of 5)

```
License information
Enter the system type that you are licensed for.
The following system types are recognized:
CLX/R
CLX800
K100
K120
K200
CYCLONE
CLX2000
K1000
K2000
S74
S740
S7000
S7400
K10000
K20000
S70000
S72000
S74000
OTHER
There is no default value for this parameter.

Please enter your selection:  s7000

Will you be installing from tape or from an archive file?
Enter TAPE or ARCHIVE.
The default value for this parameter is "TAPE"
```

Figure 3. An example PAK file installation (Part 2 of 5)

PAK file installation

```
Please enter the selection:              archive


Enter the name of the archive file to be used for installation.
The file name entered will be validated by opening it.
If the file cannot be opened you will be given the
opportunity to correct the name.
Note:if you are using an archive file,you MUST have
installed the "unpak"utility in a location that is
in the default search path for this user.
There is no default value for this parameter.

Please enter the file name:             mqnsk510

Enter the name of your spooler process.(eg: $S )
Do not enter the spooler location.
The name entered will be validated by opening it.
If the spooler cannot be opened you will be given the
opportunity to correct the name.
The default value for this parameter is "$S"
Please enter the spooler name:         $S

Select the type of installation to be performed.
The following options are available:
SCRATCH     - a from scratch installation
UPGRADE     - an upgrade from the latest service
              level of MQSeries V2.2.0.1
The default value for this parameter is "SCRATCH"


Please enter the type of installation:      scratch

Enter the volume that you will use for installation.
Enter the volume name in the format "$VVVVVVV".
The default value for this parameter is "$SYSTEM"
```

Figure 3. An example PAK file installation (Part 3 of 5)

```
Please enter the volume:      $audit1

Enter the default volume that you want Queue Managers to
be created on.
Note that the default Queue Manager volume may be changed
at any time after installation by editing the MQSINI file.
Enter the volume name in the format "$VVVVVVV".
The default value for this parameter is "$AUDIT1"

Please enter the volume:

Enter the subvolume on $AUDIT1 that you will use for executables.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSEXE"

Please enter the subvolume:

Do you wish to install the OSS pax files?
Enter YES or NO.
The default value for this parameter is "NO"

Please enter your choice:      yes

Enter the subvolume on $AUDIT1 where you want the PAX Files put.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSPAX"

Please enter the subvolume:

Select the language to be used for administration messages.
The following languages are available:
ENUS US English
ESES Spanish
FRFR French
DEDE German
ITIT Italian
JAJP Japanese
KOKR Korean
PTBR Brazilian Portuguese
ZHCN Simplified Chinese
ZHTW Traditional Chinese
The default value for this parameter is "ENUS"
Please enter the language:           enus
```

Figure 3. An example PAK file installation (Part 4 of 5)

```
License verified.

You have selected the following parameters for installation:
Archive file for installation:           MQNSK510
Spooler name:                            $S
Volume for installation:                 $AUDIT1
Default Queue Manager volume:            $AUDIT1
Subvolume for executables:               ZMQSEXE
Language for messages:                   ENUS
Install OSS PAX files to:                ZMQSPAX
This is not an upgrade to a prior V2.2.0.1 installation.
Beginning to restore files to $AUDIT1.
Ready to restore? (yes or quit):    yes

Restoring product to $AUDIT1...
Finished restoring files.
If the summary information indicates a potential error,
review the 2 spooler jobs named #instmqm, and if necessary, repeat
the installation.
Relinking native executables...
Securing files...
Creating MQSINI file...
Finished creating MQSINI file.
Creating message file...
Finished creating message file.
Installation complete.
```

Figure 3. An example PAK file installation (Part 5 of 5)

An example PAK file installation (UPGRADE installation)

If you select an UPGRADE installation, a check is made for the latest service level of MQSeries for Tandem NSK. If this check is not satisfied the installation terminates.

To upgrade a PAK file installation:

1. Unpack the installation files from $*.install subvolume on tape as follows:

```
$AUDIT1 INSTALL 78> unpak $vol.subvol.archive,$*.install.*,map names
$*.*.* to $audit1.zmqssys.*, nounload, listall, myid
```

   where *$vol.subvol.archive* is the PAK file and `$audit1.zmqssys` is the subvolume from where you want to run the INSTMQM program.

2. Run the INSTMQM program. See Figure 4 on page 105 for an example.

```
$AUDIT1 ZMQSSYS 79> instmqm
$AUDIT1 ZMQSSYS 79..


IBM MQSeries for Compaq NSK, Version 5.1
Installation and License update program.


@(#) Licensed Materials - Property of IBM  5724-A39
(C) Copyright IBM Corp. 1993, 2001 All Rights Reserved
US Government Users Restricted Rights - Use, duplication or
disclosure restricted by GSA ADP Schedule Contract with IBM Corp.


Product installation selected...
You may stop the installation by entering
"quit" at any prompt.
Where there is a default for a parameter, you may
select it by pressing the Enter key on its own.

Phase 1: Collection of license information

License information
Enter the system type that you are licensed for.
The following system types are recognized:
CLX/R
CLX800
K100
K120
K200
CYCLONE
CLX2000
K1000
K2000
S74
S740
S7000
S7400
K10000
K20000
S70000
S72000
S74000
OTHER
There is no default value for this parameter.
```

Figure 4. An example PAK file installation (UPGRADE install) (Part 1 of 4)

```
Please enter your selection:  s7000

Will you be installing from tape or from an archive file?
Enter TAPE or ARCHIVE.
The default value for this parameter is "TAPE"

Please enter the selection:              archive


Enter the name of the archive file to be used for installation.
The file name entered will be validated by opening it.
If the file cannot be opened you will be given the
opportunity to correct the name.
Note:if you are using an archive file,you MUST have
installed the "unpak"utility in a location that is
in the default search path for this user.
There is no default value for this parameter.

Please enter the file name:             mqnsk510

Enter the name of your spooler process.(eg: $S )
Do not enter the spooler location.
The name entered will be validated by opening it.
If the spooler cannot be opened you will be given the
opportunity to correct the name.
The default value for this parameter is "$S"

Please enter the spooler name:

Select the type of installation to be performed.
The following options are available:
SCRATCH     - a from scratch installation
UPGRADE     - an upgrade from the latest service
              level of MQSeries V2.2.0.1
The default value for this parameter is "SCRATCH"

Please enter the type of installation:      upgrade

Enter the volume that you installed MQSeries on.
Enter the volume name in the format "$VVVVVVV".
The default value for this parameter is "$SYSTEM"

Please enter the volume:      $audit1

Indicate the name of the MQSeries Initialization file.
This file is usually called MQSINI
Enter this in the format "$Vol.Subvol.Filename".
The default value for this parameter is "$AUDIT1.ZMQSSYS.MQSINI".
Please enter the fully qualified name of the MQSeries Initialization file.
$audit1.zmqssys.mqsini
```

Figure 4. An example PAK file installation (UPGRADE install) (Part 2 of 4)

```
Enter the subvolume on $AUDIT1 containing the MQSeries executables.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSEXE"

Please enter the subvolume:


Verifying latest service level of V2.2.0.1 is present...

Presence of PTF U473441 has been verified.
Installation proceeding.
Do you wish to install the OSS pax files?
Enter YES or NO.
The default value for this parameter is "NO"

Please enter your choice:      yes

Enter the subvolume on $AUDIT1 where you want the PAX Files put.
Enter the subvolume name in the format "VVVVVVVV".
The default value for this parameter is "ZMQSPAX"

Please enter the subvolume:

Select the language to be used for administration messages.
The following languages are available:
ENUS US English
ESES Spanish
FRFR French
DEDE German
ITIT Italian
JAJP Japanese
KOKR Korean
PTBR Brazilian Portuguese
ZHCN Simplified Chinese
ZHTW Traditional Chinese
The default value for this parameter is "ENUS"

Please enter the language:

License information updated successfully
License verified.
```

Figure 4. An example PAK file installation (UPGRADE install) (Part 3 of 4)

```
You have selected the following parameters for installation:
Archive file for installation:           MQNSK510
Spooler name:                            $S
Volume for installation:                 $AUDIT1
Subvolume for executables:               ZMQSEXE
Language for messages:                   ENUS
Install OSS PAX files to:                ZMQSPAX
This is an upgrade to a prior V2.2.0.1 installation.
The existing MQSeries Initialization file is $AUDIT1.ZMQSSYS.MQSINI
Beginning to restore files to $AUDIT1.
Ready to restore? (yes or quit):     yes

Restoring product to $AUDIT1...
Finished restoring files.
If the summary information indicates a potential error,
review the 2 spooler jobs named #instmqm, and if necessary, repeat
the installation.
Relinking native executables...
Securing files...
Finished securing files.
Updating MQSINI file =$AUDIT1.ZMQSSYS.MQSINI...
Finished updating MQSINI file $AUDIT1.ZMQSSYS.MQSINI
Creating message file...
Finished creating message file.
Installation complete.
```

Figure 4. An example PAK file installation (UPGRADE install) (Part 4 of 4)

Appendix E. Notices

This information was developed for products and services offered in the United States. IBM may not offer the products, services, or features discussed in this information 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 information. The furnishing of this information does not give 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:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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 information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information 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 United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England
SO21 2JN.

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 Programming License Agreement, or any equivalent agreement between us.

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.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, or other countries, or both:

MQSeries
MVS/ESA
IBMLink
SupportPac
IBM
FFST
BookManager

Compaq and NonStop are trademarks of Compaq Computer Corporation.

Lotus and Lotus Notes are trademarks of Lotus Development Corporation in the United States, or other countries or both.

Microsoft and Windows NT are trademarks of Microsoft Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Other company, product, and service names may be trademarks or service marks of others.

Index

Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To make comments about the functions of IBM products or systems, talk to your IBM representative or to your IBM authorized remarketer.

When you send comments 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.

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

  User Technologies Department (MP095)
  IBM United Kingdom Laboratories
  Hursley Park
  WINCHESTER,
  Hampshire
  SO21 2JN
  United Kingdom
- By fax:
  - From outside the U.K., after your international access code use 44–1962–816151
  - From within the U.K., use 01962–816151
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever method you use, ensure that you include:

- The publication title and order number
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.