

MQSeries[®] Integrator for Sun Solaris



Installation Guide

Version 2.0.2

MQSeries[®] Integrator for Sun Solaris



Installation Guide

Version 2.0.2

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix E. Notices" on page 83.

Fourth edition (June 2001)

This edition applies to IBM® MQSeries Integrator for Sun Solaris Version 2.0.2 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2000, 2001. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

| | | | |
|--|------------|---|-----------|
| Figures. | v | Installation of NEONRules and NEONFormatter Support for MQSeries Integrator | 23 |
| Tables. | vii | Installation of DB2 for Sun Solaris | 24 |
| About this book | ix | Setting up a database | 24 |
| Who this book is for | ix | Setting specific environment variables | 24 |
| What you need to know to understand this book | x | Setting up for LAN installation | 25 |
| Terms used in this book | x | Silent installation | 25 |
| | | What to do if something goes wrong during installation | 27 |
| Summary of changes | xi | | |
| Changes for this edition (GC34-5842-03) | xi | Chapter 4. Configuring a broker domain. | 29 |
| Changes for the previous edition (GC34-5842-02) | xii | Reviewing the assumptions about this configuration. | 34 |
| Changes for the second edition (GC34-5842-01) | xii | Setting up user IDs and groups | 36 |
| | | Creating and connecting to the databases | 37 |
| Chapter 1. Installation overview | 1 | Setting up database authorizations | 40 |
| Installing the runtime support | 1 | Customizing the database | 41 |
| Installing the configuration components | 2 | Configuring your broker domain | 41 |
| Post-installation configuration and verification | 2 | Checking the components | 46 |
| | | Starting your broker domain | 46 |
| Chapter 2. Planning for installation | 3 | | |
| System setup | 3 | Chapter 5. Verifying your installation | 53 |
| Hardware requirements | 3 | Preparing for verification | 53 |
| Software requirements | 5 | Creating the MQSeries resources on Sun Solaris | 53 |
| Database summary | 7 | Importing and deploying the MQSeries Integrator resources on Windows NT | 54 |
| License information | 7 | Running the predefined verification applications | 56 |
| National language support | 7 | Running the Results Service application. | 56 |
| Product components | 8 | Running the Scribble application | 58 |
| Runtime. | 8 | Running the Postcard application | 60 |
| Secondary components. | 9 | Building and using a message flow | 62 |
| Security considerations | 12 | Creating the MQSeries resources | 63 |
| Migration considerations. | 14 | Creating a message flow | 63 |
| | | Assigning the message flow to the broker | 64 |
| Chapter 3. Installing MQSeries Integrator | 15 | Deploying the message flow to the broker | 64 |
| Delivery media | 15 | Testing the message flow. | 65 |
| Preparing for installation. | 16 | | |
| Before you start. | 16 | Appendix A. System changes after installation | 67 |
| Installation procedure. | 20 | Directory structure | 67 |
| Installation of MQSeries Integrator for Sun Solaris | 22 | Environment variables | 68 |
| | | Database contents | 69 |

| | | | |
|---|-----------|--|------------|
| Default MQSeries resources | 71 | Trademarks | 85 |
| Appendix B. Setting up an Oracle8 broker database on MQSeries Integrator | 73 | Glossary of terms and abbreviations | 87 |
| Naming | 73 | Bibliography | 95 |
| Schema | 73 | MQSeries Integrator Version 2.0.2 | |
| Sizing | 73 | cross-platform publications | 95 |
| User Identifier | 74 | MQSeries Integrator Version 2.0.2 | |
| ODBC | 74 | platform-specific publications | 95 |
| Appendix C. Uninstalling MQSeries Integrator | 77 | MQSeries Everyplace publications. | 95 |
| Before you start. | 77 | NEONRules and NEONFormatter Support for | |
| Uninstalling | 77 | MQSeries Integrator publications | 96 |
| Contacting your IBM Support Center | 77 | Softcopy books | 96 |
| Appendix D. Applying maintenance | 81 | Portable Document Format (PDF) | 96 |
| Applying maintenance to MQSeries Integrator | 81 | MQSeries publications | 97 |
| Restoring a previous service level. | 82 | MQSeries Publish/Subscribe publications | 98 |
| Applying maintenance to IBM DB2 Universal | | DB2 publications | 98 |
| Database | 82 | MQSeries information available on the | |
| Appendix E. Notices. | 83 | Internet | 98 |
| | | Index | 101 |
| | | Sending your comments to IBM | 105 |

Figures

| | | | | | |
|----|--|----|----|--|----|
| 1. | Kernel parameter values - (Solaris 2.7 system). | 19 | 5. | Configuring a simple broker domain | 30 |
| 2. | Kernel parameter values - recommended by DB2 for Sun Solaris | 19 | 6. | Configuring a simple broker domain continued | 31 |
| 3. | Installation of MQSeries Integrator for Sun Solaris, part 1 | 20 | 7. | Configuring a simple broker domain continued | 32 |
| 4. | Installation of MQSeries Integrator for Sun Solaris, part 2 | 21 | 8. | Configuring a simple broker domain continued | 33 |

Tables

| | | | | | |
|----|---|----|----|--|----|
| 1. | Component disk space requirements on Sun Solaris | 4 | 5. | /var/mqsi directory structure after installation | 68 |
| 2. | Supported databases for brokers and user data | 7 | 6. | Database tables for brokers | 69 |
| 3. | Summary of authorization in the UNIX environments. | 13 | 7. | MQSeries Integrator default objects | 71 |
| 4. | /opt/mqsi/ directory structure after installation | 67 | 8. | File names of MQSeries Integrator book PDFs | 97 |

About this book

This book explains how to plan for and install IBM MQSeries Integrator for Sun Solaris. It also describes how to verify the installation.

“Chapter 1. Installation overview” on page 1 provides a high-level installation overview of MQSeries Integrator for Sun Solaris.

“Chapter 2. Planning for installation” on page 3 describes the preparation you need to complete prior to product installation.

“Chapter 3. Installing MQSeries Integrator” on page 15 provides detailed installation information for MQSeries Integrator for Sun Solaris.

“Chapter 4. Configuring a broker domain” on page 29 explains how you configure a basic broker domain.

“Chapter 5. Verifying your installation” on page 53 explains how to deploy your broker network and verify its operation using the supplied verification programs.

The appendixes cover the configuration established by the default installation options, and details of servicing and removing the product.

For further information about the product, and planning for its use, refer to the *MQSeries Integrator Introduction and Planning* book.

For details of administrative tasks, including configuration and problem determination, see the *MQSeries Integrator Administration Guide*.

A glossary and bibliography are also provided.

Who this book is for

This book is for administrators of systems on which MQSeries Integrator for Sun Solaris components will be installed and tested.

About this book

What you need to know to understand this book

To understand this book, you need to be familiar with:

- Sun Solaris system facilities.
- Windows NT[®] or Windows[®] 2000 system facilities.
- MQSeries for Sun Solaris administration facilities.
- MQSeries for Windows NT and Windows 2000 administration facilities.
- The database product that will be used to support the MQSeries Integrator for Sun Solaris components.

Terms used in this book

All references to MQSeries Integrator are to MQSeries Integrator Version 2 unless otherwise stated.

All references in this book to Windows NT are also applicable to Windows 2000 unless otherwise stated. MQSeries Integrator components that are installed and operated on Windows NT can also be installed and operated on Windows 2000.

Terms are defined in “Glossary of terms and abbreviations” on page 87.

The book uses the following shortened names:

MQSeries

A general term for IBM MQSeries messaging products.

MQSeries Publish/Subscribe

The MQSeries Publish/Subscribe SupportPac[™] that is available on the Internet for several MQSeries server operating systems. The Internet URL is given in the section “MQSeries information available on the Internet” on page 98.

DB2[®] A general term that refers to IBM DB2 Universal Database[®] Enterprise Edition, Connect Enterprise Edition, and Extended Enterprise Edition.

<mqsi_root>

A term that refers to the root directory for the product: /opt/mqsi.

Summary of changes

This section describes changes in this edition of *MQSeries Integrator for Sun Solaris Installation Guide*. Changes since the previous edition of the book are marked by vertical lines to the left of the changes.

Changes for this edition (GC34-5842-03)

The following changes have been made:

- “Chapter 2. Planning for installation” on page 3
 - Table 1 on page 4 updated.
 - “Software requirements” on page 5 updated.
 - Table 2 on page 7 added.
 - “National language support” on page 7 updated.
 - “Product components” on page 8 updated.
 - Table 3 on page 13 updated.
- “Chapter 3. Installing MQSeries Integrator” on page 15
 - “Delivery media” on page 15 updated.
 - Figure 1 on page 19 and Figure 2 on page 19 updated.
 - “Installation procedure” on page 20 updated.
- “Chapter 4. Configuring a broker domain” on page 29
 - “Setting up user IDs and groups” on page 36 added.
 - “Customizing the database” on page 41 added.
 - “Configuring the syslog on Sun Solaris” on page 44 added.
 - “Connecting the broker to the domain” on page 50 updated.
- “Chapter 5. Verifying your installation” on page 53
 - “Creating a message flow” on page 63 updated.
 - “Assigning the message flow to the broker” on page 64 updated.
- “Appendix A. System changes after installation” on page 67
 - “Environment variables” on page 68 updated.
 - Table 6 on page 69 updated.

Minor editorial changes have been made throughout the book.

Changes

Changes for the previous edition (GC34-5842-02)

The following changes have been made:

- Information about verifying the installation has been removed from “Chapter 4. Configuring a broker domain” on page 29 and put into a new, separate chapter. See “Chapter 5. Verifying your installation” on page 53.
- Migration information has been removed from this book. Refer instead to the *MQSeries Integrator Administration Guide*.
- Information about using NEONRules and NEONFormatter has been removed from this book. Refer instead to the *MQSeries Integrator Administration Guide*.

Minor editorial changes have been made throughout the book.

Changes for the second edition (GC34-5842-01)

The major changes are:

- New migration information. See “Migration considerations” on page 14.
- Information about setting specific environment variables. See “Environment variables” on page 68.
- Inclusion of details for defining the ODBC connections for Sun Solaris. See “Creating and connecting to the databases” on page 37.
- Addition of broker database support using Oracle 8. See “Appendix B. Setting up an Oracle8 broker database on MQSeries Integrator” on page 73.
- Removal of information about the components you must run on Windows NT. You should refer instead to the *MQSeries Integrator for Windows NT Version 2.0.1 Installation Guide*.
- See “Chapter 4. Configuring a broker domain” on page 29 for information about configuration and verification of all MQSeries Integrator components (on Sun Solaris and Windows NT).

Chapter 1. Installation overview

MQSeries Integrator provides comprehensive facilities to create, configure, and manage a broker domain on Sun Solaris.

A broker domain consists of:

- One or more message brokers on Sun Solaris that support diverse applications exchanging information in many formats.
The message brokers work with an optional component, the User Name Server, that provides access control for publish/subscribe applications. These components are installed together to provide the runtime support.
- Two components that provide configuration and management support:
 - The Configuration Manager owns and controls the configuration of the broker domain, the procedures (message flows or business rules) that operate within your brokers, and the definition of message formats that can be manipulated by those procedures.
 - The Control Center (on Windows NT) is a sophisticated graphical interface that allows controlled access to the resources defined to the Configuration Manager to create, change, delete, and deploy those resources, and to monitor and manage their operational status.

A full description of the components of MQSeries Integrator, the facilities they provide, and the formats of information supported, is provided in *MQSeries Integrator Introduction and Planning*.

Installing the runtime support

The runtime support (the message broker and the User Name Server) must be installed on Sun Solaris. You can install and configure one or more message brokers on one or more Sun Solaris systems, subject to your licence agreement.

You must configure the User Name Server on Sun Solaris if you wish to use topic security in a publish/subscribe environment.

“Chapter 2. Planning for installation” on page 3 provides details about:

- The hardware and software prerequisites for MQSeries Integrator runtime support
- The database support that is required by the brokers

Installing runtime support

“Chapter 3. Installing MQSeries Integrator” on page 15 tells you how to install the runtime support. You are recommended to install the runtime support before you install the configuration support on Windows NT, using the information presented in “Chapter 2. Planning for installation” on page 3, and “Chapter 3. Installing MQSeries Integrator” on page 15.

Installing the configuration components

You need to install and run the Configuration Manager and the Control Center on Windows NT. These components are supplied on the MQSeries Integrator for Windows NT product CD. Refer to the product CD and the *MQSeries Integrator for Windows NT Installation Guide* for information about the hardware and software prerequisites and the installation procedures for these components.

You need to install one Configuration Manager in your domain. You can install one or more Control Centers on one or more systems running Windows NT.

Post-installation configuration and verification

When you have installed a broker on Sun Solaris, and have installed the Configuration Manager and Control Center on Windows NT, you can verify your installation.

“Chapter 4. Configuring a broker domain” on page 29 gives instructions for configuring your broker domain to verify your installation. It also gives information on using the supplied verification programs to introduce you to some of the basic concepts and facilities of MQSeries Integrator.

Chapter 2. Planning for installation

There are three steps to installing MQSeries Integrator:

1. Planning and preparation:

Careful planning of your installation helps you to clarify your requirements and the actions you need to take setup your environment.

This chapter covers:

- a. "System setup".
- b. "Product components" on page 8.
- c. "Security considerations" on page 12.
- d. "Migration considerations" on page 14.

2. Installation:

When you have decided which components you want to install, follow the guidance in "Chapter 3. Installing MQSeries Integrator" on page 15. The installation program checks for the prerequisite products, if any, required by the components you choose.

3. Configuration:

After you have installed MQSeries Integrator, you need to carry out some configuration tasks to define and activate the resources in your installation. The tasks are described in "Chapter 4. Configuring a broker domain" on page 29.

A simple configuration is used to illustrate the tasks needed and the results of the steps taken. A set of simple tests that verify that the installation has worked is also described.

For details of the requirements for the Configuration Manager and the Control Center, see the *MQSeries Integrator for Windows NT Installation Guide*.

System setup

This section provides details of the prerequisite products for installation, and related planning and setup information.

Hardware requirements

The hardware requirements for MQSeries Integrator for Sun Solaris are listed in the following sections.

System setup

General requirements

The general hardware requirements for MQSeries Integrator for Sun Solaris are:

- MQSeries servers:
 - Any Sun SPARC desktop or server system running version 2.7 or 2.8 of the operating system.
 - Any Sun UltraSPARC desktop or server system.
- Any communications hardware supporting NetBIOS, SNA LU 6.2, SPX, or TCP/IP.
- A minimum of 512 MB of RAM to support run-time operation of components.

Disk space required

The installation requirements depend on which components you install and how much working space you need. This in turn depends primarily on your use of MQSeries resources such as queues and persistent messages.

Table 1 gives the component storage requirements in megabytes (MB).

Table 1. Component disk space requirements on Sun Solaris

| Component | MB |
|--|-------|
| Runtime (Broker and User Name Server) | 66 |
| Online documentation | 30 |
| Samples and SDK | 1 |
| NEON Interface | 46 |
| Tivoli® Interface | 3.5 |
| German NLS message catalog (De_DE) | 3 |
| Spanish NLS message catalog (Es_ES) | 3 |
| French NLS message catalog (Fr_FR) | 3 |
| Italian NLS message catalog (It_IT) | 3 |
| Japanese NLS message catalog (Ja_JP) | 3 |
| Korean NLS message catalog (Ko_KR) | 3 |
| Brazilian Portuguese NLS message catalog (Pt_BR) | 3 |
| Traditional Chinese NLS message catalog (Zh_TW) | 3 |
| Simplified Chinese NLS message catalog (Zh_CN) | 3 |
| TOTAL | 243.5 |

Software requirements

Note: Minimum supported levels are shown. Later compatible levels, if any, are supported unless otherwise stated.

Prerequisite software

The following products are prerequisites:

- Sun Solaris Version 2.7 or 2.8
- IBM MQSeries for Sun Solaris server Version 5.1 or later.
Version 5.1 must be at service level Corrective Service Diskette (CSD) 4 or later. CSD 6 is supplied in the MQSeries Integrator package. To find the current service level, use the following command:

```
pkginfo | grep mqm
```

You should look at `readme.txt` on the product CD to see the latest levels of software required. If you want to run the Scribble or Postcard samples on Sun Solaris, you need to have installed the MQSeries Client for Java™ on that machine.

The installation program checks that you have MQSeries for Sun Solaris Version 5.2 installed, or Version 5.1 at the correct service level.

If you do not have the MQSeries components you need, you are recommended to install these before you continue with MQSeries Integrator installation. You should check to see that these components are at the appropriate service level.

Note: Version 5.0 is not supported at any service level.

- A database product to support your broker or brokers. All supported databases are shown in Table 2 on page 7.

An MQSeries Integrator broker requires access to a database for internal caching and for storing internal control information. The remaining components do not need access to a database.

If one of the supported databases is installed already on your system, you can use it to support MQSeries Integrator.

Note: You must ensure that a supported database is available before you can use MQSeries Integrator.

DB2 V6.1 requires FixPack 2 U469454 (this can be found on the MQSeries Integrator for Sun Solaris supplemental CD in the `supplement_sol/db2_for_solaris_v610_fixpack2` directory).

DB2 also requires an additional 250 MB of disk storage.

System setup

Note: The supplied DB2 product has **restricted license terms and agreements**. You must only use this DB2 installation in association with your licensed use of MQSeries Integrator for message management, and only the MQSeries Integrator components can make calls to the DB2 database.

The use of a database by the MQSeries Integrator components is independent of the use of databases by your applications and message flows. You are not restricted to the databases listed here for application and data storage and retrieval.

Optional products

The following products are options, not prerequisites.

- Connectivity

The network protocols supported are SNA LU 6.2 and TCP.

For SNA connectivity:

- SunLink SNA Peer-to-Peer Version 9.1
- If token ring is used: SunLink Token Ring Interface/ SBus 3.0.2 (requires patch 102463)
- Sun TRI/P Adaptor V1.0

- Databases

For a summary of the supported databases see “Database summary” on page 7.

The NEONRules and NEONFormatter Support for MQSeries Integrator component of the broker supports message definitions and processing rules created and maintained in a number of databases.

Note: These databases are for message definitions created through the NEONFormatter only. The databases required for internal product use are listed earlier in this section.

- Application programming support

The following software compilers are supported:

- Forte C 6 (Sun WorkShop C 6)
- Forte C++ 6 (Sun WorkShop C++ 6)
- Merant Server Express V1.1
- Sun WorkShop Compiler C Version 4.2 and Version 5.0
- Sun WorkShop Compiler C++ Version 4.2 and Version 5.0

Database summary

The following table shows the supported databases.

Table 2. Supported databases for brokers and user data

| Database | AIX | HP-UX | Sun Solaris | Windows NT | Windows 2000 |
|----------------------|--------------------------------------|------------------|--------------------------------------|--------------------------------------|--------------------------------------|
| DB2 ^{1 2} | 6.1 ³ 7.1 ³ | 7.1 ³ | 6.1 ³ 7.1 ³ | 6.1 ³ 7.1 ³ | 6.1 ³ 7.1 ³ |
| Microsoft SQL Server | not applicable | not applicable | not applicable | 6.5 plus SP5a 7.0 plus SP1 | 7.0 plus SP1 2000 |
| Oracle ¹ | 8.1.6 8.1.7 | 8.1.6 8.1.7 | 8.1.6 8.1.7 | 8.1.6 8.1.7 | 8.1.6 8.1.7 |
| Sybase | 12 | not supported | 12 | 12 | 12 |

Notes:

1. DB2 6.1 and DB2 7.1 on all supported operating systems, and Oracle 8.1.6 and Oracle 8.1.7 **on Sun Solaris only**, are the only DBMS supported by MQSeries Integrator that permit a database to participate as a Resource Manager in a distributed XA transaction, and coordinated by MQSeries as the XA Transaction Manager. In MQSeries Integrator, this is referred to as supporting a globally coordinated message flow.
2. You must use DB2 for the configuration and message repository databases maintained by the Configuration Manager. No other database is supported for this purpose.
3. Please check the Readme.txt file for your product to check if a Fixpack is required.

License information

Under the terms of the MQSeries Integrator license agreement, you can install one instance of each component at any one time on any one system, with the exception of the Control Center. You can install the Control Center on multiple systems providing that each Control Center is interacting with the same single Configuration Manager. You can create multiple brokers on a single system.

National language support

MQSeries Integrator Version 2.0.2 is enabled for national language support. The user interface and message catalogs are provided in the following languages:

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

National language support

- Traditional Chinese
- US English

MQSeries Integrator Version 2.0.2 can process and construct messages in any code page for which MQSeries supports conversion to and from Unicode, on all operating systems. Supported code pages are listed in the *MQSeries Application Programming Reference*.

Note: The NEONRules and NEONFormatter Support nodes (NEONMap, NEONRulesEvaluation, and NEONTransform, and the superseded NEONFormatter and NEONRules) and graphical user interfaces are supplied in US English only

MQSeries Integrator interacts with MQSeries installed in any supported language. All languages for the MQSeries messaging products are included on the MQSeries server CD supplied with MQSeries Integrator.

All messages generated for internal intercomponent message exchange are generated in code page 1208.

DB2 Version 7.1 is fully NLS-enabled and is provided in all supported languages.

To display your current locale, execute the command `locale`; to display all the locales installed on your machine, execute the command `locale -a`. To change the locale, export the environment variables `LANG` and `LC_ALL`, ensuring that `LC_MESSAGES` is set as a result.

For further information about changing language settings, refer to the *MQSeries Integrator Administration Guide*

Product components

MQSeries Integrator for Sun Solaris has one primary component and four secondary components. A set of common files is installed with the primary component.

Runtime

When you install the Runtime component, the following two sub-components are always installed:

- Broker
- User Name Server

The broker

After installation, you can create one or more brokers on each system on which you have installed the Runtime component, subject to your license

agreement (see “License information” on page 7 for details). You can configure and activate any number of brokers on these systems, subject to system resource constraints.

Each broker requires its own queue manager. However, a single queue manager can host a single broker, and the User Name Server, but they must have been created on the same system.

Each broker requires access to a database to create and maintain internal data in tables. The tables hold information about the broker’s current configuration (for example, the message flows that are assigned to it). You are advised to use a local database server for performance reasons, although client connection to a remote DB2 server is supported. If you use a DB2 client connection, you must consider network loading and reliability because delays will significantly impact the performance in the broker domain.

The database can be:

- IBM DB2 Version 6.1 or Version 7.1
- Oracle Version 8.1.6 or Version 8.1.7
- Sybase 12

“Database summary” on page 7 provides a summary of supported databases.

The User Name Server

The User Name Server requires an MQSeries queue manager to be assigned to it. The User Name Server does not require access to a database. You are recommended to configure one User Name Server within your broker domain.

Secondary components

There are optional components that you can install in your broker domain if you choose. These are:

- Samples and Software Developers’ Kit (SDK)
- Online documentation.
- Tivoli Interface
- NLS message catalogs:
 - De_DE
 - Es_ES
 - Fr_FR
 - It_IT
 - Ja_JP
 - Ko_KR
 - Pt_BR
 - Zh_TW
 - Zh_CN

Product components

You can install these components without a previous installation of MQSeries or a database.

Samples and Software Developers' Kit (SDK)

This component comprises a set of application samples, and samples that illustrate how to use the plug-in extensions.

- Application samples

These applications illustrate the basic techniques of application programming to take advantage of the full range of MQSeries Integrator function.

- Sample programs

Sample programs are supplied in C and Java. These programs are fully operational, and both source and executables are supplied.

If you choose to use these samples and run them in your broker domain, you must ensure they are running in an environment in which MQSeries connectivity is available. Check the details of the operating systems and application programming languages supported by MQSeries clients, and by applications local to queue managers.

You can also copy and modify these examples to create your own applications, or you can add sections of their code to existing applications to exploit MQSeries Integrator function.

- The verification programs **Scribble**, **Postcard**, and **Soccer** are provided to help you test out your initial installation. These are described in “Running the predefined verification applications” on page 56.

Note: You must install the MQSeries client for Java and set various environment variables in your profile before running **Scribble** and **Postcard**.

The set of programs that make up the **Soccer** application are used in the *MQSeries Integrator Programming Guide* to illustrate the various publish/subscribe programming techniques available to your application programmers.

- Libraries and header files

Library files required for building applications are included in this component.

Headers required by applications written to the Message Queue Interface (MQI) or Application Messaging Interface (AMI) are included. Their use is illustrated in the application samples.

- Software Developers' Kit

This kit contains working examples of the plug-in extensions that you can create to enhance MQSeries Integrator. Source code is provided to illustrate the programming to use the system interfaces introduced by MQSeries

Integrator, for both message parser and message processing node. Executable code is also provided. The headers and library files required by parsers and message processing nodes are also supplied.

Parsers and processing nodes execute only on a system on which an MQSeries Integrator broker is installed.

Online documentation

Information for MQSeries Integrator is provided in the `mqs_i-docs` package, this is for online viewing using the Acrobat Reader application from Adobe. Every information unit is supplied, in US English, in Portable Document Format (PDF). A searchable library in PDF, which provides a cross book index and search facility, is also provided. You can access the documentation without installing the product. The books are in the `mqs_i_solaris_documentation` subdirectory in the root directory of the supplementary CD.

PDF files for books that have been translated are installed into the location `<mqs_i_root>/docs/book/pdf/<locale>`. An index file (in HTML format) that provides a link to each book is supplied for each language in `<mqs_i_root>/docs/book`. For more information, see “Bibliography” on page 95.

To read the documentation on Sun Solaris:

1. Install the Online documentation component.
2. Start Acrobat Reader.
3. Move to the `/panels` subdirectory.
4. Open the file `bipabsol.pdf`.

You can download a free copy of Acrobat Reader (which must be at Version 4) from the Adobe Web site at

<http://www.adobe.com>

This component can be installed on any system, including one that has no other MQSeries Integrator component installed. For example, you can choose to install one copy of the documentation on a central LAN server for all users to share.

For details of all publications supplied, see “Bibliography” on page 95.

MQSeries Integrator 2 to MQSeries Integrator 1 interface

This option enables you to install NEON runtime support on its own to allow you to run the NEON rules and formats for migration to installed brokers.

To use the NEON rules and formats you must install the NEON components. These can be found on the MQSeries Integrator for Sun Solaris CD in the `neon52` directory. You can install these components using `pkgadd`.

Product components

Tivoli Interface

This option installs Tivoli configuration files and an Adobe PDF file that enable you to run Tivoli applications, providing that you have installed the Tivoli product.

For details on how to use the package, read the supplied PDF document.

NLS message catalogs

These options install message catalogs in the following national languages:

- Brazilian Portuguese (Pt_BR)
- French (Fr_FR)
- German (De_DE)
- Italian (It_IT)
- Japanese (Ja_JP)
- Korean (Ko_KR)
- Simplified Chinese (Zh_CN)
- Spanish (Es_ES)
- Traditional Chinese (Zh_TW)

If you do not install the message catalog for your national language, the messages will default to US English.

Security considerations

Security control of MQSeries Integrator components, resources, and tasks depends on the definition of users and groups of users (*principals*) to the security subsystem of the operating system. MQSeries Integrator always creates a group **mqbrkrs** on the system on which it is installed.

Users must also have the appropriate authority to MQSeries resources (queues and queue managers) and to the databases being used by the broker. Users on Sun Solaris need appropriate authority to the MQSeries resources on Sun Solaris and to those on any remote queue managers (for example, the Configuration Manager on Windows NT).

The following table provides a summary of authorizations in the UNIX[®] environment.

Table 3. Summary of authorization in the UNIX environments

| User is... | UNIX domain |
|---|---|
| Creating broker, User Name Server | <ul style="list-style-type: none"> Member of mqbrkrs, mqm The broker or User Name Server will run under the service user ID specified on the create command in most situations: this service user ID must be used to issue the create command itself. However, when 'root' is used to issue the create command it can nominate any user to run the broker or User Name Server |
| Installing | User must be a superuser |
| Uninstalling | User must be a superuser |
| Changing broker, User Name Server | Member of mqbrkrs |
| Deleting broker, User Name Server | Member of mqbrkrs , mqm |
| Starting and stopping broker, User Name Server | Member of mqbrkrs |
| Listing broker, User Name Server | Member of mqbrkrs |
| Changing, displaying, retrieving trace information | Member of mqbrkrs |
| Running User Name Server (service user ID) | <ul style="list-style-type: none"> Member of mqbrkrs The broker or User Name Server will run under the service user ID specified in the create command |
| Running broker (MQSeries non-trusted appl) (service user ID) | <ul style="list-style-type: none"> Member of mqbrkrs The broker or User Name Server will run under the service user ID specified in the create command |
| Running broker (MQSeries trusted appl) (service user ID) | <ul style="list-style-type: none"> Service user ID must be mqm mqm must be a member of mqbrkrs |
| Clearing, joining, listing MQSeries publish/subscribe brokers | Member of mqbrkrs |
| Running publish/subscribe applications | Any user, subject to MQSeries Integrator topic and MQSeries queue access control |

You must assign users (or other groups) to these local groups to allow them to perform specific tasks.

You can find more comprehensive information in the *MQSeries Integrator Administration Guide*.

For a more general discussion of security in MQSeries Integrator see *MQSeries Integrator Introduction and Planning*.

Migration considerations

Migration considerations

For guidance on planning subsequent migration or integration of brokers, see *MQSeries Integrator Introduction and Planning*. For details of actions to take for integration and migration, see the *MQSeries Integrator Administration Guide*.

Chapter 3. Installing MQSeries Integrator

This chapter tells you how to install MQSeries Integrator for Sun Solaris.

It covers the following:

- “Delivery media”.
- “Preparing for installation” on page 16.
- “Installation procedure” on page 20.
- “Setting up for LAN installation” on page 25.
- “What to do if something goes wrong during installation” on page 27.

Delivery media

The MQSeries Integrator for Sun Solaris package includes the following:

- MQSeries Integrator for Sun Solaris Version 2.0.2.

This CD includes the following:

- MQSeries Integrator for Sun Solaris with NLS support
- Documentation for MQSeries Integrator for Sun Solaris
- NEONRules and NEONFormatter Support
- Tivoli interface support files and documentation

Note to users

Up-to-date details of the service levels required are included in the MQSeries Integrator for Sun Solaris readme.txt file on the primary product CD.

- MQSeries Integrator for Sun Solaris supplemental CD.

This CD includes the following:

- The MQSeries Integrator Version 2.0.2 documentation PDF package that can be viewed without installation.
- MQSeries classes for Java support and MQSeries classes for Java Message Service (MQSeries SupportPac MA88). You may install MA88 from this CD; however, you are recommended to download the latest version of MA88 via the MQSeries Web page listed at “MQSeries information available on the Internet” on page 98.
- MQSeries for Sun Solaris Version 5.1 CSD6.
- MQSeries for Windows NT Version 5.1 CSD6.

The CSDs are provided to enable you to upgrade existing installations of MQSeries for Sun Solaris or Windows NT Version 5.1.

Delivery media

- IBM DB2 Version 6.1 FixPack 2 U469454. This can be found in the following directory:
supplement_sol/db2_for_solaris_v610_fixpack2
- Additional product service, if required.
- DB2 Enterprise Edition for Sun Solaris Version 7.1 (two CDs).
- MQSeries for Sun Solaris Version 5.2.
This product is provided in all available national languages.
- MQSeries Clients CD.
Clients for all platforms in all available national languages are included.

Note: The following CDs are also included in the package. Refer to the *MQSeries Integrator for Windows NT Installation Guide* for information about the Windows NT components.

- MQSeries Integrator for Windows NT Version 2.0.2.
This CD includes the following:
 - MQSeries Integrator for Windows NT
 - NEONRules and NEONFormatter Support
- MQSeries for Windows NT Version 5.2.
This product is provided in all available national languages.
If you already have MQSeries for Windows NT Version 5.1 installed, you must install MQSeries for Windows NT Version 5.1 CSD4 or later. CSD6 can be found on the MQSeries Integrator for Sun Solaris supplemental CD.

For details about these products, and their use with MQSeries Integrator, see “Software requirements” on page 5.

Preparing for installation

This section informs you of the steps you must take before you install and use MQSeries Integrator for Sun Solaris.

For stand-alone machines, you can create the new user and group IDs locally. For machines administered in a network information services (NIS) domain, you can create the user and group IDs on the NIS master server machine.

Note: Users and installers of MQSeries Integrator can have their userids in NIS (network information services) groups or local groups. However, the user for whom a DB2 instance is created has to have his or her userid and home directory local to the machine containing DB2.

Before you start

Before starting to install MQSeries Integrator for Sun Solaris you:

- Must review `readme.txt`, which you can find in the root directory of the CD

- Must create a user group with the name `mqbrkrs`
To do this, type the following command:

```
groupadd mqbrkrs
```
- Must add **root** to the `mqbrkrs` group
To do this, edit the `etc/group` file. Locate the line defining the `mqbrkrs` group and type `root` after the last colon. For example:

```
mqbrkrs::42428:root
```
- Must create a user ID and add to `mqbrkrs` and `mqm`
To do this, type the following command:

```
useradd -G mqbrkrs,mqm -c "MQSeries Integrator user" <mqsi>
```

Where `<mqsi>` is the user ID and `-c "MQSeries Integrator user"` adds a descriptive comment for that user.

Note that the `useradd` command creates a locked account by default. The account should now be unlocked to enable it to be used. To do this, type the following:

```
passwd <mqsi>
```

You will be prompted to supply a new password for the user.

- Must have the appropriate authority to access your MQSeries and database resources, see Table 3 on page 13 for more information.
- Must review the machine's Kernel configuration. See "Kernel Configuration" on page 18 for the recommended settings.
- Are recommended to create and/or mount a `/var/mqsi` file system.

Resource ownership

After installation, the user ID `mqm`, in group `mqbrkrs`, owns all directories in `/var/mqsi`. All files and executables beneath `/opt/mqsi` are owned by user id `bin`, in group `bin`.

If you want to run any administration commands, for example, **mqsicreatebroker** (create broker) or **mqsisstart** (start MQSeries Integrator component), your user ID must be a member of group `mqbrkrs`.

Creating a product code file system

You can also create another file system for product code. If, for example, you do not want to have the product code installed in the `/opt/mqsi` file system because it is too small (a minimum of 160 MB is required), you can do one of two things:

1. Create a new file system and mount it as `/opt/mqsi`.

Preparing for installation

2. Create a new directory anywhere on your machine that is large enough to contain the product, and create a symbolic link from `/opt/mqsi` to this new directory. For example:

```
mkdir /bigdisk/mqsi
ln -s /bigdisk/mqsi /opt/mqsi
```

Notes:

1. Whichever of these options you pick, you **must** do it before installing the product code.
2. The file system into which the code is installed can be a remote network device, for example NFS, provided that the mount options are defined on that device to allow **setuid** programs - including root access - to be run.

Installation source

The most common way of installing this product is direct from the CD. You can also set up for installation from a shared LAN drive. This is described in "Setting up for LAN installation" on page 25.

You are guided through the installation process, and are prompted for any information required for completion.

Kernel Configuration

You are recommended to compare the values set in your Kernel configuration parameters to those listed in the following figures. If a value is shown in both figures, the value in the `/etc/system` file must be the sum of the two.

To get the appropriate recommended values for your software configuration,

1. Check the recommended values for the following products:
 - MQSeries Integrator
 - MQSeries Version 5.1 or 5.2
 - DB2 (if installed)
 - Any other software that you are running, that provides a recommended value
2. Take the highest value for each parameter and compare each one to the corresponding value set in your Kernel configuration
3. If the highest recommended value is less than the current Kernel setting, you do not need to update this parameter
4. If the highest recommended value is greater than the current Kernel setting, replace the current value with the new higher value. To change a value edit the appropriate `set parameter = value` line in the `/etc/system` file. For further information on setting up the system, see the Sun Solaris System Administration documentation and documentation for other installed software products.

Figure 1 shows the recommended Kernel parameter values on a Solaris 2.7 system.

```
set lwp_default_stksize = 0x4000
set rpcmod:svc_run_stksize = 0x4000
set shmsys:shminfo_shmmax = 4194304
set shmsys:shminfo_shmseg = 1024
set shmsys:shminfo_shmmni = 1024
set shmsys:shminfo_shmmni = 8
set semsys:seminfo_semaem = 16384
set semsys:seminfo_semmni = 1024 (semmni < semmns)
set semsys:seminfo_semmmap = 1026 (semmni +2)
set semsys:seminfo_semmns = 16384
set semsys:seminfo_semmns1 = 125
set semsys:seminfo_semopm = 100
set semsys:seminfo_semmnu = 2048
set semsys:seminfo_semume = 256
set semsys:seminfo_sevmx = 32767
set msgsys:msginfo_msgmap = 1026
```

Figure 1. Kernel parameter values - (Solaris 2.7 system)

Figure 2 shows the Kernel parameter values recommended by DB2 for Sun Solaris on a 64 - 128 MB machine.

```
set msgsys:msginfo_msgmax = 65535
set msgsys:msginfo_msgmnb = 65535
set msgsys:msginfo_msgmni = 128
set msgsys:msginfo_msgssz = 16
set msgsys:msginfo_msgtql = 256
set msgsys:msginfo_msgseg = 8192
set shmsys:shminfo_shmmax = 67108864
```

Figure 2. Kernel parameter values - recommended by DB2 for Sun Solaris

In addition, you need to increase the maximum number of concurrent open file descriptors on your system. You are recommended to set a value greater than 256.

To do this, you can use the `ulimit` command in the shell in which you are starting the broker.

Prerequisites

MQSeries Integrator for Sun Solaris installation checks for the presence of all the prerequisite software required by your installation choices. If any prerequisite is not found, you are presented with a message detailing which prerequisite is missing. You can terminate installation at this point and install the prerequisite from the relevant CD.

Preparing for installation

You are advised to check the full details of prerequisites for each component given in “Software requirements” on page 5.

Installation procedure

This section describes the installation of MQSeries Integrator for Sun Solaris Version 2.0.2.

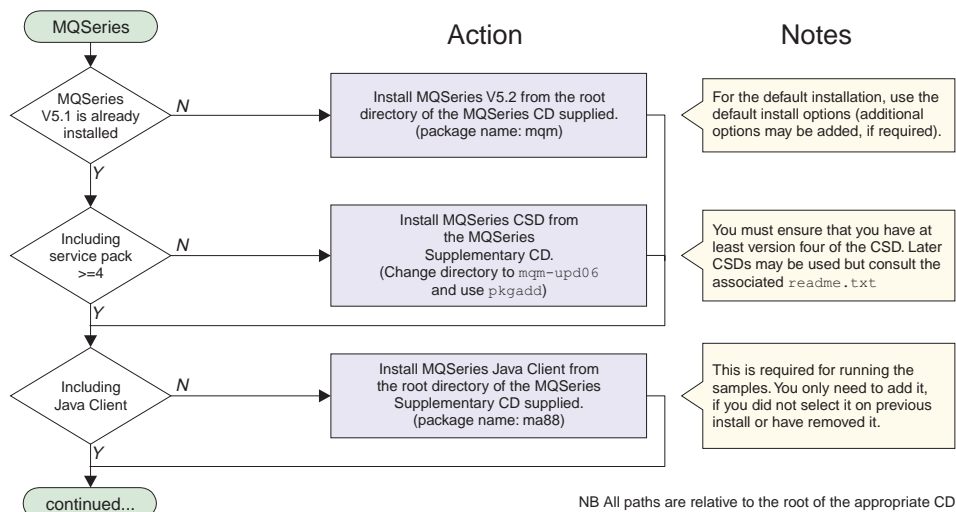


Figure 3. Installation of MQSeries Integrator for Sun Solaris, part 1

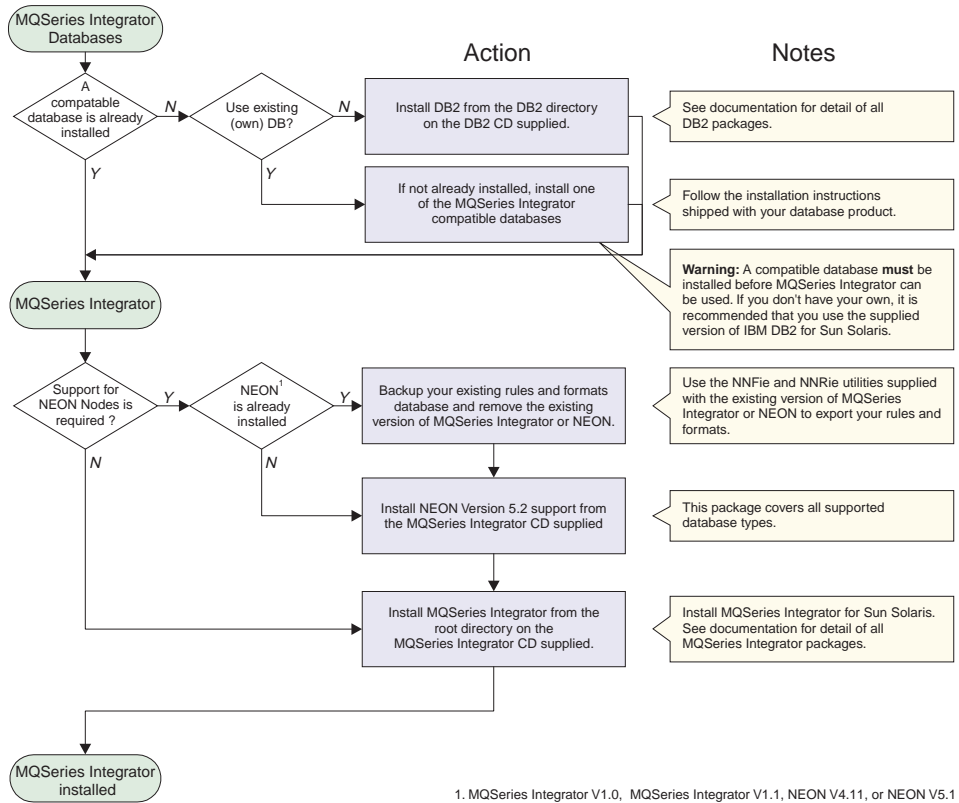


Figure 4. Installation of MQSeries Integrator for Sun Solaris, part 2

Installation procedure

Installation of MQSeries Integrator for Sun Solaris

To install MQSeries Integrator for Sun Solaris, carry out the following procedure:

1. Check to see if the Volume Manager is running on your system by typing the following command:

```
/usr/bin/ps -ef | /bin/grep vold
```

If it is running, the CD is mounted on `/cdrom/mqsi_solaris` automatically. If it is not running, mount the CD by typing the following commands:

```
mkdir -p /cdrom/mqsi_solaris  
mount -F hsfs -r /dev/dsk/cntndnsn /cdrom/mqsi_solaris
```

Where `cntndnsn` is the name of your CD-ROM device (*n* is a digit in the range 0-9).

Alternatively, use the `volcheck` command to automount a CD-ROM device.

2. Use the Solaris **pkgadd** program to install the software by carrying out the following procedure:
 - a. To install, log in as root and execute:

```
pkgadd -d <cdrom mount-point>
```

Note: If you want to create a logfile which contains a transcript of everything that appears on the screen during installation, enter the following to the command:

```
pkgadd -d /cdrom/mqsi_solaris/. 2>&1 | tee output_log_file
```

- b. Select one of the following packages:
 - mqsi MQSeries Integrator Version 2.0.2 for Sun Solaris (SPARC_SOL2.7) Version 2, Revision 2
 - mqsi-docs MQSeries Integrator Version 2.0.2 for Sun Solaris (SPARC_SOL2.7) Documentation Version 2, Revision 2
- c. Selecting the first package from the list above brings up the following:
 - 1) MQSI Runtime
 - 2) MQSI NEON Interface
 - 3) MQSI Tivoli Interface
 - 4) SDK & Samples
 - 5) Spanish message catalog
 - 6) French message catalog
 - 7) German message catalog
 - 8) Japanese message catalog
 - 9) Italian message catalog
 - 10) Brazilian Portuguese catalog
 - 11) Traditional Chinese catalog
 - 12) Simplified Chinese catalog

13) Korean catalog

Select the package you require.

- d. Press the Enter key.

During installation MQSeries Integrator for Sun Solaris checks that the following prerequisites, among others, have been installed:

- MQSeries for Sun Solaris Version 5.1 or Version 5.2
- MQSeries for Sun Solaris Version 5.1 CSD4 (if Version 5.1 is installed)

It also checks that the following have been created:

- mqm user
- mqm group
- mqbrkrs group

If any of the above are missing or not created the installation process warns you, and gives you the option to stop, but continues with the installation if you require.

Installation of NEONRules and NEONFormatter Support for MQSeries Integrator

To install the NEONRules and NEONFormatter Support, carry out the following procedure:

1. Check to see if the Volume Manager is running on your system by typing the following command:

```
/usr/bin/ps -ef | /bin/grep vold
```

If it is running, the CD is mounted on `/cdrom/mqsi_solaris` automatically. If it is not running, mount the CD by typing the following commands:

```
mkdir -p /cdrom/mqsi_solaris
mount -F hsfs -r /dev/dsk/cntndnsn /cdrom/mqsi_solaris
```

Where `cntndnsn` is the name of your CD-ROM device (*n* is a digit in the range 0-9).

Alternatively, use the `volcheck` command to automount a CD-ROM device.

2. Use the Solaris **pkgadd** program to install the software by carrying out the following procedure:
 - a. To install, log in as root and execute:


```
pkgadd -d <cdrom mount-point>/neon52
```
 - b. Select all the packages in this directory.
 - c. Press the Enter key.

Installation procedure

Installation of DB2 for Sun Solaris

To install DB2 Enterprise Edition for Sun Solaris, run the `db2install` script followed by the `db2setup` script. Both scripts can be found in the `db2_v7.1` directory.

Setting up a database

After installation you must set up your required database for use with MQSeries Integrator. There is a script provided called `mqs_i_setupdatabase` in the `/opt/mqsi/bin` directory, this will accept a database type and home directory of a database as arguments. For example, to set up DB2 for use with MQSeries Integrator the command would be:

```
mqs_i_setupdatabase db2 /opt/IBMDB2/V7.1
```

where `/opt/IBMDB2/V7.1` is the directory in which DB2 is installed.

Other supported database types are **oracle** (for Oracle Version 8) and **sybase12**.

- For further information about setting up a DB2 database, see “Chapter 4. Configuring a broker domain” on page 29.
- For further information about setting up an Oracle database, see “Appendix B. Setting up an Oracle8 broker database on MQSeries Integrator” on page 73.
- For further information about setting up a Sybase database, see the *MQSeries Integrator Administration Guide* and Sybase documentation.

If the required libraries cannot be found at the specified location the following error message will be displayed:

```
<libraries> not found in lib directory of <database home directory>  
but links created anyway
```

This might happen if the product is not accessible because it is installed on a mounted directory. The links created will **NEVER** be added to the package database and you will have to manually remove them from `/var/mqsi/lib`.

Setting specific environment variables

View the `readme.txt` file, `readme.txt`, to ensure you have the latest profile shipped with this product:

```
/opt/mqsi/sample/profiles/profile.sol
```

You must run the profile, or a copy of it, before attempting to use the product.

Setting up for LAN installation

On Sun Solaris, use one of the following two methods to make the MQSeries Integrator installation files accessible on a LAN server:

- Make the MQSeries Integrator CD-ROM drive shareable.
- Copy the product files from the CD to the server as follows:
 1. Create a directory on the LAN server to store the installation files. For example:


```
mkdir /instmqsi
```
 2. Copy the complete contents of the CD to the new directory. For example:


```
cp -rf /cdrom/mqsi_solaris/. /instmqsi
```
 3. Give all licensed users access to the directory that now contains the CD image.
 4. Make the filesystem shareable:
 - Use the **share** command. For example, to give all users read-only access using NFS:


```
share -F nfs -o ro -d "MQSeries Integrator Lan install" /instmqsi
exportfs -a
```
 5. On the target machine, create a directory which will mount the shared drive. For example:


```
mkdir /remotemqsiimage
```
 6. From a command prompt on the target machine:
 - Connect to the appropriate drive and folder using the **mount** command. For example:


```
mount <machine name>:/instmqsi /remotemqsiimage
```

Where <machine name> is the name of the target machine.

7. Change to the installation directory, which in this example is `remotemqsiimage`.
8. Enter `pkgadd -d <package_name>`

Where <package_name> is the name of the package to install. You can install `mqsi` or `mqsi-docs`.
9. Press the Enter key and follow the installation prompts.

Silent installation

MQSeries Integrator supports silent installation, where no user interaction is expected. This method of installation is typically used where you have to install many identical copies of the same software across many machines. These machines can be remote to the installer.

Silent installation

Notes to users

1. This option should only be used by experienced administrators because no checks are performed on the system.
2. The output from the installation either appears on your terminal, or can be redirected to a file that you select.

The basic command is:

```
pkgadd -r RESPONSE -a ADMIN -d /cdrom/mqsi_solaris/. mqsi
```

where an example RESPONSE file is:

```
CLASSES=MQSI-Neon MQSI-Runtime MQSI-Tivoli SDK&Samples
```

and an example ADMIN file is:

```
mail=  
instance=nocheck  
partial=nocheck  
runlevel=nocheck  
idepend=nocheck  
rdepend=nocheck  
space=nocheck  
setuid=nocheck  
conflict=nocheck  
action=nocheck  
basedir=default
```

The RESPONSE file can be generated using the following command (from root):

```
pkgask -r RESPONSE -d /cdrom/mqsi_solaris/. mqsi
```

Or, for the documentation package:

```
pkgask -r RESPONSE -d /cdrom/mqsi_solaris/. mqsi-docs
```

The ADMIN file is the file `/var/sadm/install/admin/default`, altered so that no interaction or checking is done.

Note: Both the ADMIN and RESPONSE files must be in the current directory where the command is to be run.

An example script file is supplied on the product CD in the directory called `silent`. This script, “`silent.sh`”, enables administrators to install the product silently.

```
# Copyright IBM 2001  
#  
# This script installs MQSeries Integrator Version 2.0.2 with no  
# checking of pre requisites, space etc and no user
```



```
# interaction.  
pkgadd -r RESPONSE -a ADMIN -d /cd_rom/mqsi_solaris/. mqsi
```

What to do if something goes wrong during installation

If you encounter any problems during installation, you are advised to check the following:

- Review the `readme.txt` file supplied on the CD. This has the most up-to-date information available for product installation and operation. There might be last minute changes to the installation process that you must follow. You might also find additional information on the MQSeries Web site (the address is given in “MQSeries information available on the Internet” on page 98).
- If a message is displayed with the MQSeries Integrator prefix of BIP, check the *MQSeries Integrator Messages* book to determine the cause of the error and the action you need to take to correct it.
- During installation a list of all the files installed is appended to `/var/sadm/install/contents`. Files that encountered problems during installation are shown with an exclamation mark (!) next to them.

When you have identified and corrected the error, or errors, you can run the installation program again. If this does not work, you are advised to follow the steps for manual uninstallation to ensure that your system is in a consistent state before you retry.

If you are unable to resolve the problems you have, after checking the possible sources of error listed above, you must contact your IBM Support Center. See “Contacting your IBM Support Center” on page 77 for further information.

Chapter 4. Configuring a broker domain

This chapter takes you through the tasks you must complete, on Sun Solaris and Windows NT, to set up the minimum resources required in the broker domain, after you have installed MQSeries Integrator for Sun Solaris.

There are seven tasks:

1. "Reviewing the assumptions about this configuration" on page 34.
2. "Setting up user IDs and groups" on page 36
3. "Creating and connecting to the databases" on page 37.
4. "Setting up database authorizations" on page 40.
5. "Configuring your broker domain" on page 41.
6. "Checking the components" on page 46.
7. "Starting your broker domain" on page 46.

These tasks are shown diagrammatically, starting with Figure 5 on page 30. You might find it helpful to review these figures before you start and use them to check off the tasks as you complete them. You need to complete all the steps in each task. The figures and the associated text describe the tasks to be carried out on each platform in a logical sequence. You might prefer to carry out all the tasks on Sun Solaris first, followed by all the tasks on Windows NT; the outcome is the same.

Note: SupportPac IC02 supplies a utility that simplifies the configuring of the broker (in UNIX environments) by providing an interactive Korn Shell script that ensures the associated database, queue managers, queues, and channels are set up correctly.

Configuring a broker domain

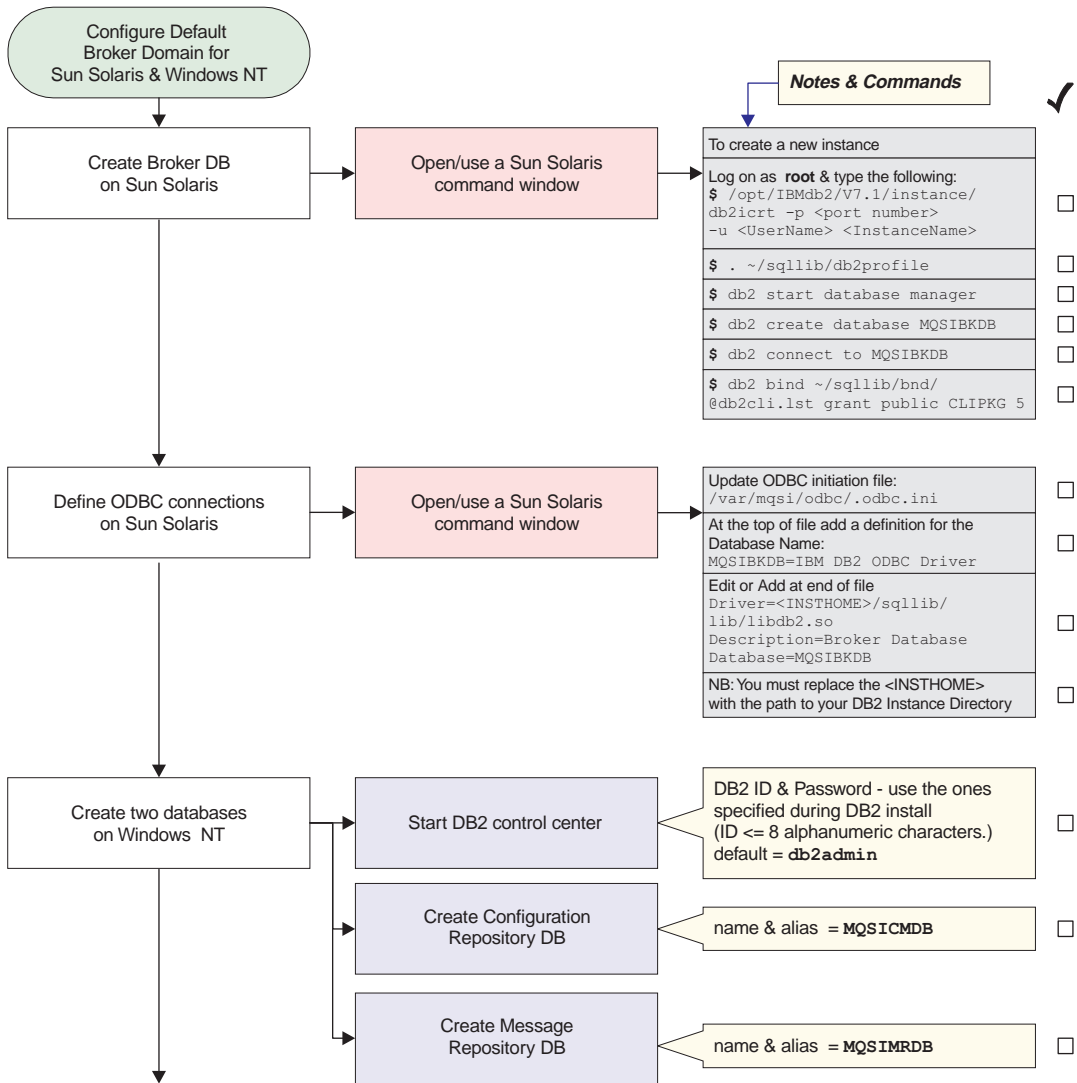


Figure 5. Configuring a simple broker domain

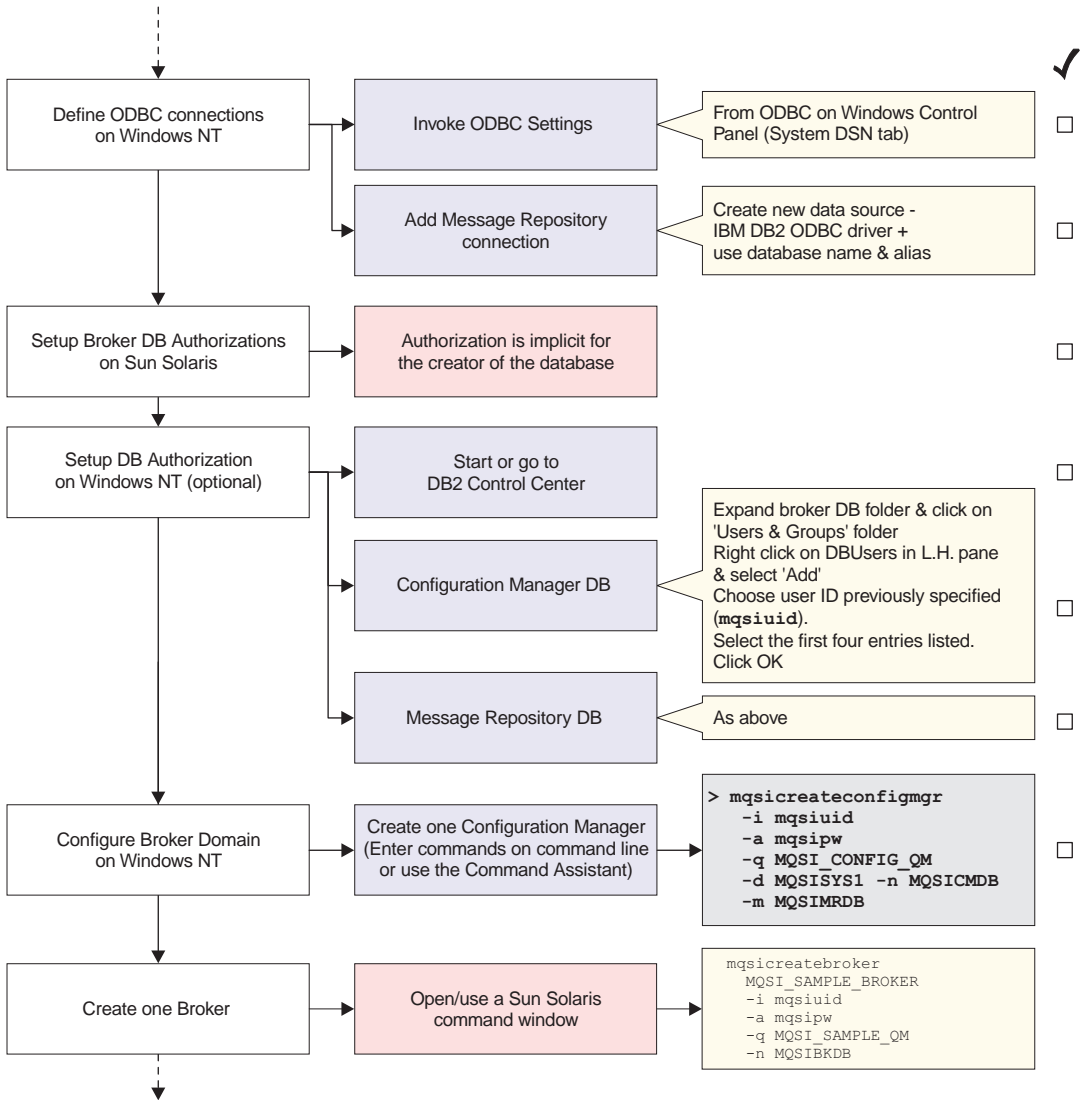


Figure 6. Configuring a simple broker domain continued

Configuring a broker domain

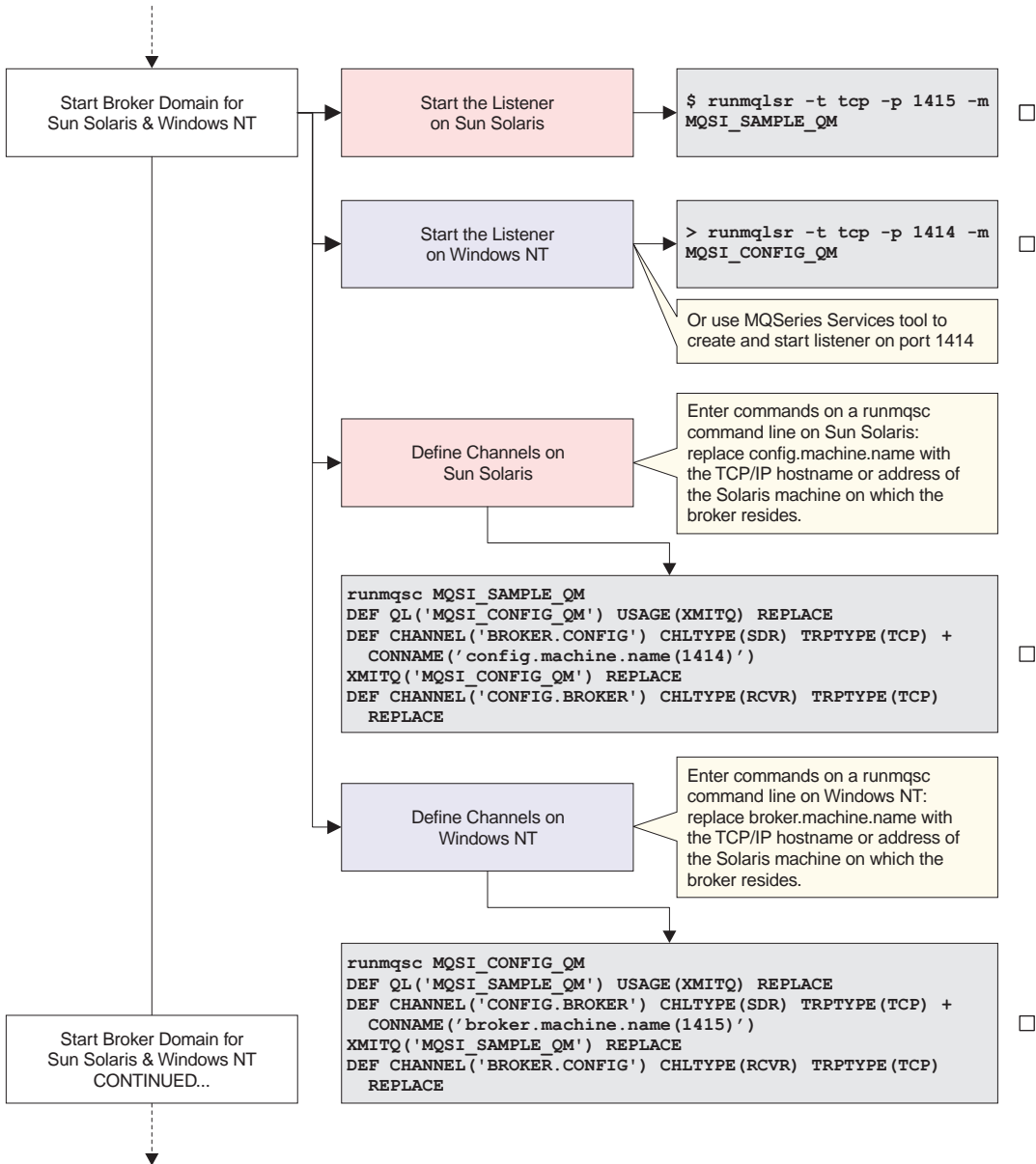


Figure 7. Configuring a simple broker domain continued

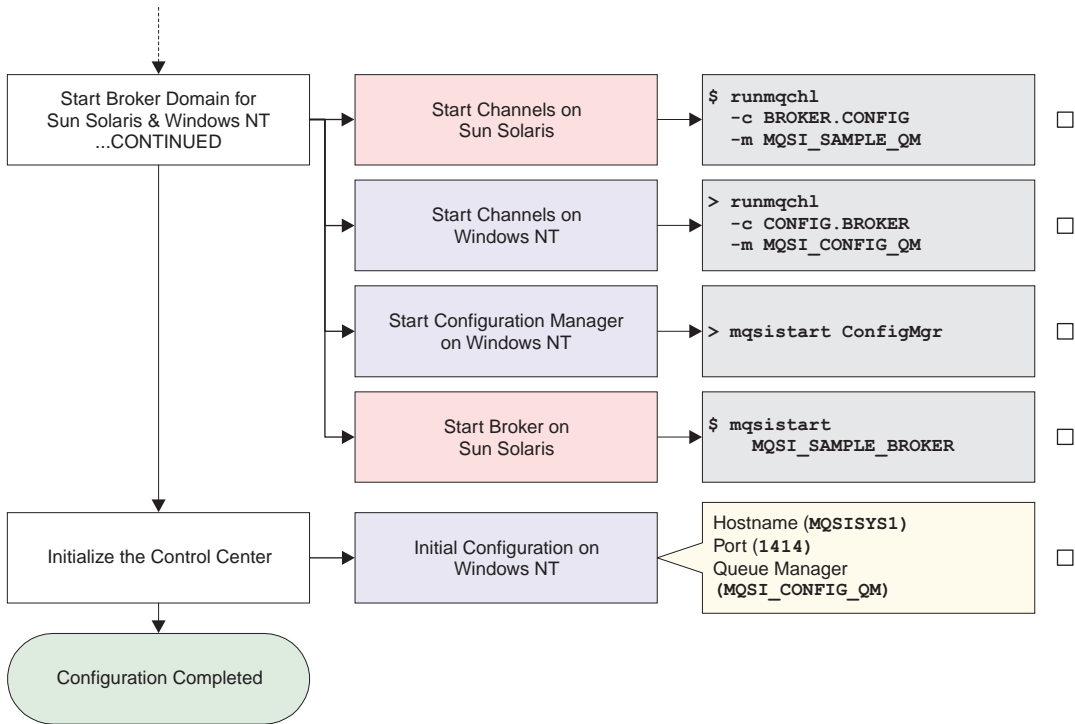


Figure 8. Configuring a simple broker domain continued

Most of these steps make use of commands supplied by MQSeries Integrator. Some of these commands can be invoked using the MQSeries Integrator Command Assistant; all of them can be invoked from the command line. The commands are given in full; you can choose which method you want to use to issue these commands.

Note: You must install the MQSeries client for Java in order to use the Command Assistant on Sun Solaris.

The Command Assistant and the configuration commands are described in detail in the *MQSeries Integrator Administration Guide*, which provides further reference and guidance material, and describes the actions you must take if you experience any errors in completing the tasks illustrated here.

Reviewing the assumptions

Reviewing the assumptions about this configuration

Before you start to define any resources, review the assumptions made about the configuration that is created. If you want to understand more about MQSeries Integrator configuration in general, refer to *MQSeries Integrator Introduction and Planning*.

The assumptions for this configuration include resource names and user IDs. If you want to override any of the assumptions, make a note of changes you want to make and apply those changes as you complete the tasks illustrated. For example, the names used for the broker and its queue manager are for illustration only. You are recommended to follow any existing naming conventions you have for MQSeries (or any other) resources. See *MQSeries Integrator Introduction and Planning* for more information about defining a naming convention.

This chapter assumes that:

- You have installed the Runtime component on Sun Solaris.
- You have installed the Configuration Manager and the Control Center components on Windows NT.
- You have installed the product using the TCP/IP hostnames MQSISYS1 on Windows NT and MQSISYS2 on Sun Solaris. You **must** replace these names wherever they are used with the hostnames of your systems, if they are different.
- The ports 1414 and 1415 are available for MQSeries. If these ports are not available, you must replace these ports wherever they are used with the ports you are using.
- The local systems MQSISYS1 and MQSISYS2 define the security domain relevant to this configuration (that is, all users and groups are defined in the local account security domain).

Note: This illustrates a very basic security scenario. You can find more comprehensive security information in *MQSeries Integrator Introduction and Planning*, and more complex scenarios illustrated in the *MQSeries Integrator Administration Guide*.

- You are logged on with the same user ID that you used to install MQSeries Integrator. If you are not, you must ensure that your current logon ID is a member of the following:
 - The mqbrkrs group on Sun Solaris
 - The **Administrator** group on Windows NT
- You have authorized the user IDs between Sun Solaris and Windows NT. You must ensure that:

- The Serviceuserid that you will use on Windows NT (specified on the `mqsicreateconfigmgr` command) is defined on the machine running the Sun Solaris broker.
- The Serviceuserid that you will use on Sun Solaris (specified on the `mqsicreatebroker` command) is defined on the machine running the Configuration Manager.

In this example the service user ID is `mqsuid`. If you have chosen another service user ID, you must define this on both Sun Solaris and Windows NT. See the *MQSeries Integrator Administration Guide* for more information about setting up user IDs and groups.

- You have decided to create a new user ID, the 'service userid' (*mqsuid*). This will be used as the user ID for running MQSeries Integrator services (the Configuration Manager and the broker) and accessing MQSeries Integrator databases.

In most cases, you cannot change user IDs after your configuration has been set up, so you are advised to check where you use them very carefully.

- The configuration consists of a broker, installed on Sun Solaris, and the Configuration Manager, installed on Windows NT, communicating through MQSeries.
- A set of sample names and other default values are used for MQSeries Integrator, MQSeries, and database resources. You can use the sample names and defaults exactly as they are shown, or you can decide to use your own names, to follow the naming conventions you have in place. If you choose to use your own names, you **must** change the names and default values to match your configuration, whenever they are used in the tasks illustrated.
- You are using DB2 Version 6.1 or later for all database requirements. DB2 has been installed as part of your installation procedure, and you have since restarted your system and verified your DB2 installation.

After you have verified your installation, and understood and implemented the basic principles of operating your broker domain, you will probably need a more complex MQSeries network for your broker domain. Your brokers and the Configuration Manager will be located on different physical machines, and you are likely to include a User Name Server in your broker domain. As you distribute components of MQSeries Integrator you will have to set up users and groups on Sun Solaris and Windows NT and authorize them between systems. For more detailed guidance and instructions, refer to the *MQSeries Integrator Administration Guide*.

Setting up user IDs and groups

Setting up user IDs and groups

If you are creating a new user ID for MQSeries Integrator for Sun Solaris see “Before you start” on page 16.

If you are creating a new user ID for MQSeries Integrator for Windows NT, invoke the Windows NT User Manager to create the user ID and assign it to all the MQSeries Integrator groups. If you are using an existing user ID, invoke the User Manager to assign the user ID to the MQSeries Integrator groups. You must authorize a user ID to run the Configuration Manager on Windows NT.

Note: If you choose to create a new User ID, do not give it the same name as your machine, otherwise the following error occurs: BIP8026 Unable to start the component.

You must ensure that the user ID is a member of the Windows NT group **Administrators**. This is independent of the user ID you define in the MQSeries Integrator groups to run MQSeries Integrator components. Membership of **Administrators** provides the level of security required by the MQSeries operations completed here.

Be careful to specify the correct user IDs wherever they appear in this chapter. You can define and use the sample user ID defined here. If you decide to use a different (new or existing) user ID, be sure to use it wherever the sample user ID is shown in the following tasks.

Take the following steps:

Step 1. Invoke the Windows NT User Manager. You can access this program from the Windows NT Start menu (the default is *Start* —> *Programs* —> *Administrative Tools* —> *User Manager*).

Step 2. Create a new user ID.

- a. Select the *User* menu and select *New user...*
- b. Fill in the fields on the dialog presented to create the user ID `mqsuid`, with password `mqsipw`.
Note that the maximum length for the `mqsuid` field is 12 characters.

Step 3. Add the new user ID to the groups required for running the components.

- a. Click the **Groups** button on the *New User* dialog. This presents another dialog, *Group Memberships*.
- b. Add the new user to the MQSeries Integrator group **mqbrkrs**, to the Windows NT group **Administrators**, and to the MQSeries group **mqm**.

- c. Click **OK**. The User Manager returns you to the *New User* dialog.
- d. Click **OK**.

Step 4. Add your logon ID to the groups that authorize you to complete all the tasks in the Control Center.

- a. Double click your current user ID. This presents the *User properties* dialog.
- b. Click the **Groups** button on this dialog. This presents another dialog, *Group Memberships*.
- c. Add the new user to **mqbrasgn**, **mqbrdevt**, **mqbroops**, and **mqbrtpic**.
- d. Click **OK**. The User Manager returns you to the *User Properties* dialog.
- e. Click **OK**.

Step 5. You can now close the User Manager.

Creating and connecting to the databases

On any machine running DB2 you must bind the db2cli package to the configuration-manager database.

You need to complete the steps in the following sections:

Creating and connecting to the database on Sun Solaris

If you ran db2setup, a default instance was created. You are recommended to use the default instance, but if you want to create a separate instance, use the command in step 1. Otherwise, start at step 2.

Note: If you create a separate instance, the InstanceName must match an existing user ID. You need to create a user ID if one does not exist already. The user and the user's home directory should be local to the machine containing DB2 and should not be NIS or NFS mounted.

1. Log on as root and enter the following command:

```
/opt/IBMd2/V7.1/instance/db2icrt -u <UserName> <InstanceName>
```

Where <InstanceName> is the name of the new instance.

2. Log on as <username> using the following command:

```
. ~/sqllib/db2profile  
db2 start database manager  
db2 create database MQSIBKDB  
db2 connect to MQSIBKDB  
db2 bind ~/sqllib/bnd/@db2cli.lst grant public CLIPKG 5
```

Note that `. ~/sqllib/db2profile` should be in the profile of all users. See "Setting specific environment variables" on page 24 for more information.

Creating and connecting to the databases

3. To enable MQSeries Integrator brokers to use the database, you must add definitions for the database to the ODBC initiation file (`/var/mqsi/odbc/.odbc.ini`).

Edit the file and add the following lines.

- At the top of the file add a definition for the Database Name:

```
MQSIBKDB=IBM DB2 ODBC Driver
```

- At the end of the file, edit or add the following lines:

```
Driver=<INSTHOME>/sql1lib/lib/libdb2.so  
Description=Broker Database  
Database=MQSIBKDB
```

The path identified by the Driver definition is specific to your installation, so you must replace the `<INSTHOME>` with the path to your DB2 Instance directory.

Note: Problems with ODBC can be traced by modifying the `/var/mqsi/odbc/.odbc.ini` file as follows:

```
[ODBC]  
Trace=1  
TraceFile=/var/mqsi/odbc/odbctrace.out  
TraceDll=/opt/mqsi/merant/lib/odbctrac.so  
InstallDir=/opt/mqsi/merant
```

The userid that MQSeries Integrator runs under must have write access to this file.

If you are using Oracle8 to support the broker, see “Appendix B. Setting up an Oracle8 broker database on MQSeries Integrator” on page 73.

Creating and connecting to the database on Windows NT

1. Start the DB2 Control Center from the Start menu (*Start* → *Programs* → *DB2 for Windows NT* → *Control Center*).

If you are using DB2 Version 6.1 you must enter a valid user ID and password on the Sign On dialog presented.

- Use the DB2 Administrator ID you specified when you installed DB2. The default user ID is `db2admin`. If you changed this, enter the user ID you specified.
- Enter the password for the user ID you are using.

2. Create the databases.

Expand the Object tree in the DB2 Control Center until you find Databases. Right-click Databases and select *Create Database using Smartguide*.

Note: DB2 database names are limited to eight characters.

Creating and connecting to the databases

When you have created the database (or databases) click **Done**. A confirmation message indicating successful completion of the create command appears at the bottom of the window.

You are recommended to create two databases:

- The configuration repository.
Enter the name and alias of your database. This chapter assumes that MQSICMDB is used for both the name and the alias. If you choose a different name, use it here and remember to use it in all other steps that refer to this database.
- The message repository.
Enter the name and alias of your database. This chapter assumes that MQSIMRDB is used for both the name and the alias. If you choose a different name, use it here and remember to use it in all other steps that refer to this database.

If you prefer, you can create a single database to hold all the tables required. Whichever scheme you choose, ensure that you use the correct name whenever you are asked to specify a database in subsequent commands.

If you wish to use the DB2 command line processor, bind the db2cli package to the configuration-manager database, MQSICMDB for example, by using the following commands:

```
db2 connect to MQSICMDB
db2 bind c:\sqllib\bnd\@db2cli.lst blocking all grant public
db2 connect reset
```

where c:\ is the drive on which you installed DB2.

3. Define the ODBC connections.
 - a. From the Windows NT Start menu, select *Start* → *Settings* → *Control Panel*.
 - b. Within the Control Panel, double-click the ODBC icon (labeled *ODBC* or *ODBC Datasources*).
 - c. Click the *System DSN* tab.
 - d. You must add an ODBC connection for the message repository. The configuration repository does not need an ODBC connection if it is created as a separate database. However, if you have created a database that will be used for the message repository as well as the configuration repository, you must create an ODBC connection for that database.
 - 1) Click the Add button. The *Create New Data Source* window appears.
 - 2) Double-click IBM DB2 ODBC DRIVER.

Creating and connecting to the databases

- 3) Choose the data source (database) name from the dropdown list.
- 4) Click **OK**.

When you have completed these steps for the message repository database, click **OK**.

Setting up database authorizations

- On Sun Solaris

You do not need to setup broker database authorizations on Sun Solaris because authorization is implicit for the creator of the database.

- On Windows NT

You need to authorize selected user IDs to access the database (or databases) you have created, to enable the MQSeries Integrator resources to operate successfully.

If you wish to use the DB2 command line to complete this step, issue the appropriate DB2 grant commands. If you wish to use the DB2 Control Center, the steps required are shown below. If you need further guidance about any of these steps, use the Control Center's online help facility.

Note: You can omit these steps if you choose to specify your DB2 administrator ID and password for the datasource and database IDs and passwords in the Configuration Manager command. This option is not illustrated here. See the *MQSeries Integrator Administration Guide* for further information.

1. Start the DB2 Control Center, if it is not already active. Log on with the DB2 administrator user ID you used in "Creating and connecting to the databases" on page 37.
2. Complete the following:
 - a. Expand the object tree until you find the database.
 - b. Expand the tree under this database and click the *User and Group Objects* folder. The *DB Users* and *DB Groups* folders are displayed in the right pane.
 - c. Right-click the *DB Users* folder in the right pane and select *Add* from the pop-up menu. The Add User notebook opens.
 - d. Select the user ID `mqsuid` (or the ID you are using for MQSeries Integrator database access) from the dropdown list.
Select the appropriate options in the box labelled *Choose the appropriate authorities to grant to the selected user* for all the databases you have created for MQSeries Integrator.

The ID you specify as the `ServiceUserID` on the create commands, `mqsuid` (or the user ID you are using in place of this sample ID), must have the following authority to the database you created for MQSeries Integrator:

- Connect database
 - Create tables
 - Create packages
 - Register functions to execute in database manager's process
- e. Click **OK**. The authority or authorities are granted. The dialog is closed.

3. Close the DB2 Control Center.

Customizing the database

You are recommended to increase the database heap size to at least 900 (4 KB pages). Enter the following DB2 command, replacing <dbname> with the name of the database you wish to customize:

```
db2 update database configuration for <dbname> using dbheap 900
```

On Windows NT, this operation can also be carried out using the DB2 Control Center. Expand the object tree until you find the database, then right click on the database name. From the pop up menu, select *Configure* —> *Performance* —> *Database heap size*. The default value is typically 600 (4 KB pages). You should choose a value of at least 900.

Configuring your broker domain

Now you are ready to define the components that make up the simple configuration by:

1. Creating a Configuration Manager on Windows NT.
2. Creating a broker on Sun Solaris.

The steps required on Windows NT are described using the Command Assistant. When you enter values in the entry fields, the command that is generated is displayed on the lower part of the screen.

The MQSeries Integrator commands are given for both platforms. On Sun Solaris, enter the commands at the command line. On Windows NT, enter the commands at a command prompt window. You are recommended to enter the commands from the \bin subdirectory of the directory in which you installed MQSeries Integrator for Windows NT (the home directory). If you accepted the default, the home directory is:

```
C:\Program Files\IBM\MQSeries Integrator 2.0
```

Creating a Configuration Manager on Windows NT

To use the Command Assistant to create the Configuration Manager:

1. Select *Start* —> *Programs* —> *IBM MQSeries Integrator 2.0* —> *Command Assistant* —> *Create Configuration Manager*.

Configuring your broker domain

2. Fill in the Service User ID field (mqsiuid), the Service Password field (mqsipw), the Queue Manager Name field (MQSI_CONFIG_QM), and the Security Domain field (MQSISYS1). Check that the values are correct and click **Next**.
3. Fill in the ConfigMgr Database Name field (MQSICMDB) and the MRM ODBC Data Source Name field (MQSIMRDB). Check that the values are correct and click **Next**.
4. Check that the command is correct and click **Finish**

Alternatively, you can enter the command directly, for example:

```
mqsicreateconfigmgr -i mqsiuid -a mqsipw -q MQSI_CONFIG_QM  
-d MQSISYS1 -n MQSICMDB -m MQSIMRDB
```

where:

- -q is the queue manager that hosts the Configuration Manager
- -i is the service user ID for running the Configuration Manager (as a Windows NT service)
- -a is the password for the service user ID
- -d is the security domain within which user authority is checked (in this case, the local account security domain defined by the hostname of the system)
- -n is the database for the configuration repository
- -m is the datasource name for the message repository

Note: The queue manager is created if it does not already exist.

The service user ID and password are also used as the user ID and password for both the configuration repository and the message repository. If you are using a different user ID and password for access to these repositories, you must specify these here (using flags -u and -p for the configuration repository and flags -e and -r for the message repository).

When you type the password, it appears on the command line exactly as you type it. However, when you type it into the Command Assistant, and when it is stored in the Windows NT registry, it is displayed as asterisks for security reasons.

If you are using different names or values for any parameter on this command, you **must** replace the appropriate values with your own.

The command might take a short while to complete. The command generates the following expected responses, unless you are using the Command Assistant:

Configuring your broker domain

The service is called *IBM MQSeries Broker ConfigMgr*. It has a default startup status of manual: you can change this to automatic if required.

- The queue manager `MQSI_CONFIG_QM` has been created and started. You can check the existence and status of this queue manager using MQSeries Services from the Start menu (*Start* → *Programs* → *IBM MQSeries* → *MQSeries Services*).
- The MQSeries resources required by the Configuration Manager have been defined on the queue manager. These resources are detailed in “Default MQSeries resources” on page 71.
- The authorizations required by the Configuration Manager to access MQSeries resources have been set (the *setmqaut* messages seen in the responses to the command).
- The database tables required by the configuration repository have been set up in the database `MQSICMDB`.
- The database tables required by the message repository have been set up in the database `MQSIMRDB`.
- The Windows NT registry has been updated to record the Configuration Manager creation.

Configuring the syslog on Sun Solaris

Before creating a broker on Sun Solaris, you are recommended to configure syslog to write all ‘user’ messages to a file, because all MQSeries Integrator messages, not generated by the command line utilities, are written to the syslog. You can select any location for the file where you want to log the messages.

To configure the syslog you need to (as root) edit the file `/etc/syslog.conf` because this file contains definitions on where to write messages written to the syslog.

All MQSeries Integrator messages are written to the ‘user’ facility, so you need to add a line starting with the text “user” and then select what level of messages you want to see. For example you can choose information, debug, warning, or error. Debug is selected in the example below.

Initially, you are advised to send all levels of message to a file. However, as you become used to the system, you may want to direct high severity message to a different file or even have them mailed to an administrator.

To direct all user facility messages to the file `/var/adm/user.log`, create the file `syslog.user` and add the following line to the end of the `syslog.conf` file:
`user.debug /var/adm/user.log`.

Before committing the changes by restarting `syslogd` you must log in as root and create the following file:

```
touch /var/adm/user.log
chown root:mqbrkrs /var/adm/user.log
chmod 750 /var/adm/user.log
```

You must restart the syslog daemon (as root) for your changes to take effect.

Creating a broker on Sun Solaris

To create a broker, enter the following command:

```
mqsicreatebroker MQSI_SAMPLE_BROKER -i mqsiuid -a mqsipw
-q MQSI_SAMPLE_QM -n MQSIBKDB
```

where:

- MQSI_SAMPLE_BROKER is the broker
- -q is the broker's queue manager
- -n is the database that has been created for the broker tables
- -i is the service user ID for the broker
- -a is the password for the broker

The service user ID and password are also used as the user ID and password for the broker database. If you want to use a different user ID (using flag -u) and password (flag -p) to access this database, you must specify them here and grant the user ID access to the database (described in "Setting up database authorizations" on page 40).

When you enter the password on the command line, it appears on the screen exactly as you type it.

If you are using different names or values for any parameter in this command, you **must** replace the appropriate values with your own.

The command might take a short while to complete. The expected responses generated by this command are:

```
MQSeries queue manager created.
Creating or replacing default objects for MQSI_SAMPLE_QM.
Default objects statistics : 29 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
MQSeries queue manager 'MQSI_SAMPLE_QM' started.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
BIP8071I: Successful command completion.
```

Configuring your broker domain

If the command detects any errors, or is unable to complete, it returns an error message on the command line, or in the syslog, which includes the explanation and action in full. The error might have been caused by another component that MQSeries Integrator interacts with to complete this command (DB2 or MQSeries), so check for errors from these products too.

On completion:

- The broker MQSI_SAMPLE_BROKER has been created.
- The queue manager MQSI_SAMPLE_QM has been created and started.
- The MQSeries resources required by the broker have been defined. These resources are detailed in “Default MQSeries resources” on page 71.
- The required authorizations for the MQSeries resources have been set (the *setmqaut* messages seen in the responses to the command).
- The database tables required by the broker have been set up in the database MQSIBKDB. These tables are listed in Table 6 on page 69.

Checking the components

You can check the existence of the MQSeries Integrator components you have created on each platform using the `mqsilist` command.

- On a Sun Solaris command line type:

```
mqsilist
```

This displays the broker component, with the queue manager which supports it. Now that the broker has been created, the command responds with:

```
BIP8099I: MQSI_SAMPLE_BROKER - MQSI_SAMPLE_QM
```

```
BIP8071I: Successful command completion.
```

- At a Windows NT command prompt type:

```
mqsilist
```

This displays the Configuration Manager component, with the queue manager that supports it. Now that the Configuration Manager has been created, the command responds with:

```
BIP8099I: ConfigMgr - MQSI_CONFIG_QM
```

```
BIP8071I: Successful command completion.
```

Starting your broker domain

When you have created the MQSeries Integrator components, you can start to activate your broker domain. You must have Windows NT **Administrator** authority on your Windows NT machine to complete these steps.

You are recommended to complete the tasks described here in the following order:

1. Starting the MQSeries listeners.
2. Defining the channels.
3. Starting the channels.
4. Starting the Configuration Manager on Windows NT.
5. Starting the broker on Sun Solaris.
6. Starting the Control Center on Windows NT.

The MQSeries Integrator commands are used to illustrate these steps, with sufficient information provided to complete the task. For a full description of these commands, and possible errors, see the *MQSeries Integrator Administration Guide*. (You cannot use the MQSeries Integrator for Windows NT Command Assistant for these commands.) If errors are reported, you can also check the Application view of the Windows NT Event Viewer and the Sun Solaris syslog.

Starting the MQSeries listeners

Most of the resources you need to support this configuration have already been created and started for you when you invoked the create broker and create Configuration Manager commands. There is just one extra step you need to take to enable the Control Center to communicate with the Configuration Manager.

- On a Sun Solaris command line type:

```
runmq1sr -t tcp -p 1415 -m MQSI_SAMPLE_QM
```
- On Windows NT use one of the following two methods:
 - You are recommended to use MQSeries Services (*Start* → *Programs* → *IBM MQSeries* → *MQSeries Services*). Expand the left-hand pane and find and click the queue manager (MQSI_CONFIG_QM) to display its services in the right-hand pane. If the Listener is listed, right-click the Listener, and select *All Tasks* → *Start*. This starts the listener as a background task.

If the Listener is not listed, right-click the queue manager and select *New* → *Listener*. This creates a listener with default properties of transport type TCP and port 1414. When it has been created, right-click the Listener and select *Start*.

This starts the listener as a background task.
 - Enter the following command on the command line:

```
runmq1sr -t tcp -p 1414 -m MQSI_CONFIG_QM
```

The listener is started as a foreground task and is not displayed in the MQSeries Services window.

Note: If the default MQSeries port 1414 is not available (perhaps because it is already in use by another queue manager), you must assign an

Starting your broker domain

alternative port number. The port value must be set in the Listener properties dialog (on the Parameters tab), or as the `-p` parameter on the `runmq1sr` command. If the port is already in use, the Control Center cannot contact the Configuration Manager. For example, if you have set up a default queue manager on this system, it probably already has a listener started on this port. You can check if any listeners are already active using MQSeries Services.

Defining the channels

Use the following examples to define the channels on each platform:

- On Sun Solaris

Run the example below on a `runmqsc` command line on the Solaris machine:

```
runmqsc MQSI_SAMPLE_QM
DEF QL('MQSI_CONFIG_QM') USAGE(XMITQ) REPLACE
DEF CHANNEL('BROKER.CONFIG') CHLTYPE(SDR) TRPTYPE(TCP) +
  CONNAME('config.machine.name (1414)') +
  XMITQ('MQSI_CONFIG_QM') REPLACE
DEF CHANNEL('CONFIG.BROKER') CHLTYPE(RCVR) TRPTYPE(TCP) REPLACE
```

- On Windows NT

Run this example on a `runmqsc` command line on the Windows NT machine:

```
runmqsc MQSI_CONFIG_QM
DEF QL('MQSI_SAMPLE_QM') USAGE(XMITQ) REPLACE
DEF CHANNEL('CONFIG.BROKER') CHLTYPE(SDR) TRPTYPE(TCP) +
  CONNAME('broker.machine.name (1415)') +
  XMITQ('MQSI_SAMPLE_QM') REPLACE
DEF CHANNEL('BROKER.CONFIG') CHLTYPE(RCVR) TRPTYPE(TCP) REPLACE
```

Starting the channels

Enter the following commands to start the sender channel on each platform:

- On Sun Solaris:

```
runmqchl -c BROKER.CONFIG -m MQSI_SAMPLE_QM
```

- On Windows NT:

```
runmqchl -c CONFIG.BROKER -m MQSI_CONFIG_QM
```

Starting the Configuration Manager on Windows NT

Start your Configuration Manager by entering the following command on the command line. You cannot do this using the Command Assistant.

```
mqsisstart configmgr
```

This command initiates the start up of the Configuration Manager's Windows NT service and can only report on whether that service is started successfully. If it is, you must check the Application view of the Windows NT Event Viewer to ensure that the Configuration Manager has initialized successfully.

The Configuration Manager cannot contact a broker until a reference to that broker exists in the configuration repository. You must create this reference to the broker using the Control Center. The steps required to do this are described in “Preparing for verification” on page 53.

Starting the broker on Sun Solaris

Start your broker by entering the following command on the command line. If you are not using the sample broker name, substitute your broker name for MQSI_SAMPLE_BROKER in the command.

```
mqsistart MQSI_SAMPLE_BROKER
```

You must check the syslog to ensure that the broker has initialized successfully.

Starting the Control Center on Windows NT

Start the Control Center by double-clicking the Control Center icon in the MQSeries Integrator program folder, or by using the Windows NT Start menu (*Start* → *Programs* → *IBM MQSeries Integrator 2.0* → *Control Center*).

Complete the following tasks to set up the environment you need to complete the simple verification described in “Chapter 5. Verifying your installation” on page 53.

This section gives only a minimum of information required to complete your initial broker domain setup. For further information about the Control Center, refer to *MQSeries Integrator Using the Control Center*.

Complete the initial dialog presented by the Control Center, *Configuration Manager Connection*, to provide the information needed to connect your Control Center session to the Configuration Manager. The fields are:

1. **Hostname.** This is initially blank. Enter the network hostname of the system on which the Configuration Manager has been created. In the simple configuration defined in this chapter, the value you must enter here is MQSISYS1. If you are using a different host name, enter your value here.
2. **Port.** This is initially blank. Enter the number of the port on which the queue manager is listening. You set this up in “Starting the MQSeries listeners” on page 47. The default port is 1414.
3. **Queue Manager name.** This is initially blank. Enter the name of the queue manager (MQSI_CONFIG_QM). This queue manager already has a definition for the server connection required by the Control Center (the channel SYSTEM.BKR.CONFIG of type SVRCONN), which was created when the Configuration Manager was created.

Starting your broker domain

When you have completed these fields, click **OK**. The Control Center now contacts the Configuration Manager, which might take a few minutes. If the Control Center fails to make contact, the most likely reasons are:

- The Configuration Manager has not started successfully.
- The listener has not started successfully.
- The queue manager is not available.
- You are logged on to the local security domain, but this user ID is not a member of the MQSeries Integrator groups. Check the groups of which your current user ID is a member. Also, check that you are logged on to the same security domain as the one you were logged on to when you installed MQSeries Integrator.

Check for MQSeries or MQSeries Integrator entries in the Windows NT Event log (Application view) to track down the problem.

If you want to check, or change, these settings at a later time, click *File* —> *Connection* to bring up the connection dialog.

Connecting the broker to the domain

By connecting the broker to the domain, you are adding the definitions required to the Configuration Manager so that it will be able to define flows to the broker. This relies on the Configuration queue manager being able to route messages to the Broker queue manager using the channels defined.

To add a broker to the domain:

1. Using the Control Center select the Topology panel. In the left-hand panel, click with the right mouse button on the Topology icon and select Check out. A key symbol should appear next to the Topology icon.
2. In the right-hand pane, click with the right mouse button and select *Create* —> *Broker*. This opens a dialog box in which you enter the name of your broker on Sun Solaris, 'MQSI_SAMPLE_BROKER', and the name of the Broker queue manager, 'MQSI_SAMPLE_QM'. Select Finish when complete.

When complete, an icon identifying the broker will be created in the right hand panel. However, at this point neither the Configuration Manager nor the broker has been contacted.

3. Click with the right mouse button on the Topology icon in the right-hand pane and select Check in. When complete, the key symbol disappears from the Topology icon.

At this stage the broker definition has been stored in the Configuration Manager database but nothing has been sent to the broker. To verify the connection between the Configuration Manager and broker you should run the default execution process against the broker.

To deploy the empty default execution group to the broker:

1. Select the Assignments tab. In the furthest right-hand pane you will see a definition for the broker, and within the broker icon an empty box containing the word default.

This indicates that there is one execution group defined for the broker and that the execution group contains no definitions.

2. In the far-left panel, click with the right mouse button on the icon identifying your broker and select *Deploy* → *Complete Assignments Configuration*.

This attempts to send the default execution group definition to the broker. After a short delay, a dialog box is displayed, indicating that the deploy request has been submitted to the Configuration Manager.

3. To see the results of the deploy request you must select the Log panel.

All deployed responses are written to this panel. To refresh the display, click the Refresh icon in the top left-hand corner of the panel.

If the response is successful, the broker on your Sun Solaris machine will start two new processes. You will see 'DataFlowEngine' and bipservice MQSI_SAMPLE_BROKER in response to `ps -ef`

If the response was not successful you should use the messages generated, and errors written, to the Windows Event Log (Applications View) to determine the cause of the error.

You have now:

- Created a Configuration Manager on Windows NT
- Created a broker on Sun Solaris
- Added the broker to the broker domain defined by the Configuration Manager

See “Chapter 5. Verifying your installation” on page 53 for information on creating message flows for you to deploy for the Sun Solaris broker and test using the sample applications.

Chapter 5. Verifying your installation

You have now completed the configuration and activation tasks. This chapter explains how to deploy your broker domain, and how to verify your installation. You can choose to run one or more of a set of verification programs that illustrate different aspects of setup and operation:

- “Preparing for verification”.
- “Running the predefined verification applications” on page 56.
 - “Running the Results Service application” on page 56.
 - “Running the Scribble application” on page 58.
 - “Running the Postcard application” on page 60.
- “Building and using a message flow” on page 62.

All the tasks illustrated here assume you have used the sample names and values when you completed the tasks in “Chapter 4. Configuring a broker domain” on page 29. If you have changed any of these names or values, make sure that you use your values in this section.

You complete most of the tasks involved in running these verification programs using the Control Center. This section gives the minimum information you need to complete these tasks. For further information, see *MQSeries Integrator Using the Control Center*.

Preparing for verification

Before you can run any of the verification programs, you must complete some preparation.

Creating the MQSeries resources on Sun Solaris

The verification applications require local queues on the broker’s queue manager. This step creates the MQSeries queues needed by the applications. The queues are:

- For the Results Service application:
 - MQSI_SOCCER_PUBLICATION_QUEUE
 - MQSI_SOCCER_SUBSCRIPTION_QUEUE
- For the Scribble application:
 - MQSI_SCRIBBLE_PUBLICATION_QUEUE
 - MQSI_SCRIBBLE_SUBSCRIPTION_QUEUE
- For the Postcard application:
 - MQSI_POSTCARD_INPUT_QUEUE
 - MQSI_POSTCARD_OUTPUT_QUEUE

Preparing for verification

An MQSC command file is provided to define these resources. The file is in the `sample/mqsc` subdirectory under the MQSeries Integrator home directory. From the command line change to this directory and type the following:

```
runmqsc MQSI_SAMPLE_QM < samples.tst
```

Importing and deploying the MQSeries Integrator resources on Windows NT

You must now work with the MQSeries Integrator resources that are used by the applications.

1. Close the Control Center. Stop the Configuration Manager using the command:

```
mqsisstop configmgr
```

2. Import the message set required by the Postcard application into the message repository. The message set is defined in the file `PostcardMS.mrp`. To import the message set, change to the directory `examples\postcard` in the MQSeries Integrator home directory. Enter the following command:

```
mqsiimpexpmsgset -i -n MQSIMRDB -u mqsiuid -p mqsipw -f PostcardMS.mrp
```

Note: Use the same user ID and password in this command that you specified for message repository access when you created the Configuration Manager (flags `-e` and `-r`).

For more details of this command, see the *MQSeries Integrator Administration Guide*.

3. Restart the Configuration Manager (`mqsisstart configmgr`). The Configuration Manager can now access the new message set and make it available.
4. Restart the Control Center and select the *Topology* view. Check out the broker domain topology by selecting the topology, right-clicking, and selecting *Check Out*. This locks the topology and allows you to make changes to it.
5. The title bar currently displays *Untitled* to show that you have an empty workspace. Import the supplied workspace import file that defines the resources used by the verification programs.
 - a. Select *File* → *Import to Workspace*. The Import dialog is displayed. This allows you to select the type of resources you want to import, and the file that contains the resource definitions.
 - b. The valid resource types to import are:
 - Message flows
 - Topics
 - Topology

The file supplied by MQSeries Integrator contains message flow and topology definitions, so you must select these two types.

- c. Click **Browse** and locate the \examples subdirectory (in the MQSeries Integrator home directory). Select the sample workspace import file SamplesWorkspaceForImport, click **Select**, and then click **Import**.

The definitions can take a few minutes to import. When import has finished, a message dialog is displayed, confirming that the resources have been imported successfully. Click **OK** to close the dialog. You now see the sample broker (MQSI_SAMPLE_BROKER) in the topology.

- d. Select the *Message Flows* view. Check that the import has created two new folders of message flows, *Verification message flows* and *IBM Default message flows*. The default message flows are described in *MQSeries Integrator Introduction and Planning*. The default message flow folders are in addition to the *IBMPrimitives* folder. If you expand the tree for the verification message flows you can see three new message flows, one for each of the verification programs. They are *ScribbleInversion*, *Soccer*, and *Postcard*.
6. Check that the message flows have already been assigned to the broker's default execution group (this happened when you imported the workspace); they should be showing in the Domain hierarchy and Topology panels.
7. Save the changes that you have made. Select *File* → *Check in* → *All (Save to Shared)*. The contents of the configuration repository are updated with the new definitions and assignments and everything is checked into the repository.
8. Deploy your changes to the broker. When you deploy, the Configuration Manager sends information to the broker about the resources it needs to support the message flow services.
 - a. Select the *Topology* view.
 - b. Select *File* → *Deploy* → *Complete Configuration (all types)* → *Normal*, or right-click the Topology root and select *Deploy*.
A message dialog confirms initiation. Select **OK** to dismiss the dialog.
9. Select the *Log* view and refresh the contents by clicking the green refresh icon. It can take a few minutes for all the deployment messages and responses flowing between the Configuration Manager and the broker to be displayed. Keep refreshing this view until you see the completion messages. If everything is successful, the log will contain the following messages: BIP4045I, BIP4040I, and BIP2056I.
10. View the deployed configuration graphically in the *Operations view*. Refresh the view, and the topology view is displayed.

Running the predefined applications

Running the predefined verification applications

This section describes how to run each of the three applications supplied with MQSeries Integrator. You can run any of these, in any order, immediately after installation or at any time in the future. If you choose to run these applications later, make sure you have your system set up in the same way as the system you configured in “Chapter 4. Configuring a broker domain” on page 29 (or make the appropriate adjustments as you follow these steps).

The verification applications also illustrate how MQSeries Integrator can be used to transform and route messages outside the programming logic of the participating applications, which can therefore run unaffected by updates to that transformation logic, or routing logic, or both.

Note: If you are not using MQSI_SAMPLE_QM as the queue manager for your broker, you must change the broker queue manager name in the output nodes of the verification sample programs.

Running the Results Service application

The Results Service application is written in the C programming language and demonstrates a number of basic publish/subscribe features. The application is a simple implementation of a soccer match results gathering service. It consists of one or more publisher applications, and one subscriber application. You can find the files that make up this application (source, header files, and executables) in the `sample/soccer` directory.

Run this application on Sun Solaris as follows:

1. Start the subscriber application **soccerResults**.

You must start a single subscriber that subscribes to all soccer matches being played, and displays the results for them. The subscriber application functions as a results server. You must start **soccerResults** before you start any instances of the publisher application, so that the results server does not miss any publications.

Start the Results Service as follows:

- a. Change to the `/opt/mqsi/sample/soccer/bin` directory.
- b. Enter the command

```
soccerResults MQSI_SAMPLE_QM
```

The results server displays a message confirming that it has registered a subscription and started successfully, and you can now start the match simulator (publisher).

2. Start the publisher application, **soccerGame**.

You can run one or more publishers. Each instance publishes event publications on a single soccer match. You must specify two soccer teams as input to `soccerGame`.

Running the predefined applications

Start the publisher application as follows:

- a. Change to the `/opt/mqsi/sample/soccer/bin` directory.
- b. Enter the command to start a soccer game. Use the “_” character to represent a space in the name of a team:

```
soccerGame Team1 Team2 MQSI_SAMPLE_QM
```

```
soccergame Arsenal Manchester_United MQSI_SAMPLE_QM
```

Note: Team names can contain only the characters `_`, `0-9`, `a-z`, and `A-Z`.

How the Results Service works

The Results Service application uses messages that have a standard MQSeries header, an MQRFH2 header, and a string that specifies the playing teams and their scores.

The soccer simulator **soccerGame** publishes an event publication following this message template to the queue `MQSI_SOCCER_PUBLICATION_QUEUE` on the broker’s queue manager. The MQInput node in the *Soccer* message flow has been set up so that it identifies this queue as its input queue.

The input node retrieves the publication from this queue and forwards it to the publication node. The publication can indicate that:

- A match has started.
- A goal has been scored.
- A match has ended.

The results server **soccerResults** subscribes to all these event publications arriving on queue `MQSI_SOCCER_SUBSCRIPTION_QUEUE`. It processes these messages and displays the information: the start of a new game, a score update, and the end of a game.

One important feature of the soccer simulator **soccerGame** is its ability to maintain a current state of all the matches being played (the multiple publishers). It achieves this by publishing a retained publication message to the broker with the latest score of each match every time the score changes. This means you can restart the results server after a failure, and the results server subscribes to all these retained publications to restore the current match state to the state it had the last time the results server was running.

If you want to see this use of retained publications, you can start several instances of the publisher application **soccerGame**. When these are running,

Running the predefined applications

and a couple of goals have been scored, change to the window running the results server application and prematurely stop that process¹ using Ctrl-C.

Wait about 30 seconds, then restart the results server **soccerResults**. You will see that the matches being played are restored to their last known score, and updated by any remaining match changes that occurred whilst the results service was stopped.

If you restart the results server too quickly, it might fail to open the subscriber queue with reason 2042 (MQRC_OBJECT_IN_USE). This is because the queue manager has not yet recognized that the application has failed, and has therefore not released the queue that the application opened exclusively. You can retry the restart after a few seconds; once the queue is available it will succeed.

See the *MQSeries Integrator Programming Guide* for more details about the implementation of this application and the publish/subscribe techniques it uses.

Running the Scribble application

The Scribble application is written in Java and demonstrates a number of basic publish/subscribe and message transformation features. In contrast to the Results Service which works with multiple publishers and one subscriber, Scribble works with one publisher and any number of subscribers.

The publisher publishes the current coordinates of the line being drawn in its window, and each subscriber receives the inverted coordinates and displays the resulting drawing in its window.

You must have:

- MQSeries Client for Java on the Sun Solaris machine.
- The Efix as described in the `readme.txt` (if you are using MQSeries for Sun Solaris, V5.1).
- Environment variables correctly set in the profile. Refer to `readme.txt` for information on these.

Run this application on Sun Solaris by following these steps:

1. Start the publisher application.
 - a. Open a command window and change to the `/opt/mqsi/sample/scribble` subdirectory
 - b. Execute the `ScribblePublisher.sh` script

1. If you have created your broker to run as an MQSeries trusted application, you must not end this application in this way, because the queue will not be released. For more information about MQSeries Integrator and MQSeries trusted applications, see *MQSeries Integrator Introduction and Planning*.

Running the predefined applications

You now see a dialog that prompts you for the broker queue manager name. Enter `MQSI_SAMPLE_QM` and click **OK**. A confirmation dialog, *Scribble ready*, is displayed. Click **OK**. The publisher window is displayed.

2. Start the subscriber application.
 - a. Open a command window and change to the `/opt/mqsi/sample/scribble` subdirectory
 - b. Execute the `ScribbleSubscriber.sh` script

You now see a dialog that prompts you for the queue manager name. Enter `MQSI_SAMPLE_QM`. The dialog also allows you to enter a queue name. If you want to use the default subscriber queue, `MQSI_SCRIBBLE_SUBSCRIPTION_QUEUE`, you do not have to specify this. If you are using a different queue, you must define that queue and enter the name here. Click **OK**. The subscriber window is displayed.

3. Start dragging the mouse with either mouse button depressed to draw lines in the publisher window. These lines appear inverted in your subscriber window.

You can start multiple Scribble subscribers, but you must specify a different queue for each one. The definitions you completed in “Creating the MQSeries resources on Sun Solaris” on page 53 contains a single subscriber queue for this application, the default subscriber queue `MQSI_SCRIBBLE_SUBSCRIPTION_QUEUE`. If you want to start additional subscribers, define additional queues like the default one, and enter the queue name as well as the queue manager name at the dialog you see when you first start the subscriber.

If you have a pre-1.1.8 version of Java on your machine, follow the steps below to start Scribble:

1. Change to the `/opt/mqsi/jre/bin` directory
2. Enter the command `jre -classpath "$CLASSPATH" Scribble &`

How Scribble works

The Scribble application uses messages that have a standard MQSeries header, an MQRFH2 header, and a message body formatted in XML that specifies the drawing coordinates. When you drag the mouse across the publisher window with a mouse button depressed, it draws a line that the publisher records as a set of coordinates. It publishes each set of coordinates in an XML message to the publication queue `MQSI_SCRIBBLE_PUBLICATION_QUEUE`. The `MQInput` node in the *ScribbleInversion* message flow has been set up so that it identifies this queue as its input queue.

The input node retrieves the publication from this queue and the message flow inverts the drawing by manipulating the coordinates (transformation),

Running the predefined applications

and publishes the resulting drawing (routing) to each ScribbleSubscriber's subscription queue (MQSI_SCRIBBLE_SUBSCRIPTION_QUEUE is the default).

The details of the transformation and routing performed by the *ScribbleInversion* message flow are:

- Receive the published message in the MQInput node.
- Filter on the publish/subscribe topic *scribble/coord* in the **FilterOnTopic** node. This node is of primitive type Filter.
If the match is successful, the message is passed to the **InvertCoordinates** node, which is of primitive type Compute, for transformation.
If no match is found, the message is sent directly to the publication node without inversion.

You can see the ESQL code that implements the inversion of the coordinates in the message. Select the *Message Flows* view in the Control Center, select *ScribbleInversion* in the tree in the left pane, click the **InvertCoordinates** node with the right mouse button to display the node's context menu, and select *Properties*.

Running the Postcard application

The Postcard application is based on the MQSeries for Windows NT and Windows 2000 Postcard application, and has been extended to demonstrate the transformation capabilities of MQSeries Integrator. It is written in C with a Java end-user interface.

Postcard allows you to send a postcard to another nickname, either known to this instance, or to a different instance, of the application. You must run this application on the same system as the broker. You can find the files that make up this application (source, header files and executables) in the *sample/postcard* subdirectory under the MQSeries Integrator home directory.

You must have:

- MQSeries Client for Java on the Sun Solaris machine.
- The Efix as described in the *readme.txt* (if you are using MQSeries for Sun Solaris, V5.1).
- Environment variables correctly set in the profile. Refer to *readme.txt* for information on these.

Run this application by following these steps:

1. Start the first Postcard application.

This first instance acts as the sending application. Change directory to */opt/mqsi/sample/postcard* and execute the *postcard.sh* script.

Running the predefined applications

You now see a dialog that prompts you for a nickname to use for sending or receiving messages. Enter an alphanumeric string of up to 24 characters. The dialog also asks you for the name of the broker queue manager. This is an optional field, but if you do not enter a queue manager name, the default queue manager is used. Enter `MQSI_SAMPLE_QM` to specify the correct queue manager for verification. Click **OK**. A Postcard window appears.

2. Start the second Postcard application.

This second instance acts as the receiving application. Enter a second nickname and the name of the broker queue manager. Click **OK**. A second Postcard window appears.

3. Fill in the postcard and send the message.

In the sender (first) Postcard window, fill in the **To** field with the nickname of the receiving (second) postcard application. Fill in the remaining fields from the pulldown menus to build the content of the postcard (location, length of stay, and weather). Click the **Send** button. You see the message, marked *Sent*, in the box called “Postcards sent and received (transformed)” in the lower part of the Postcard window.

4. View the received postcard message.

You will see the postcard arrive in the second (receiving) Postcard application. Select the received message in the list and click **View** to see the contents of the (received and transformed) message. The message has been transformed to include the country of the city from which the message was sent.

5. Return a postcard to the original sender.

From the second Postcard application, select a message from the “Postcards sent and received (transformed)” box and click **Reply**. This gives you a new postcard to fill in, with the first application’s nickname already in the **To** field. Fill in the remaining fields and click **Send** to send the new postcard to the first application.

How Postcard works

The Postcard application sends messages to, and receives messages from, the *Postcard* message flow. You deployed this message flow to the broker in “Preparing for verification” on page 53. You also stored the message set *PostcardMS* in the message repository, and deployed it to the same broker. The message set is referenced by the message processing nodes within the flow. It contains one message, *PostcardMessage*, that defines these elements:

1. **Location** (of type STRING)
2. **Country** (STRING)
3. **MessageText** (STRING)
4. **Duration** (INTEGER)
5. **Recipient** (STRING)
6. **Sender** (STRING)

Running the predefined applications

7. **GoodTime** (INTEGER)
8. **Weather** (STRING)

The elements of type STRING each have an associated element length that defines the maximum number of characters valid in this element.

The application program creates and interprets the messages based on a structure defined in the C header file `postcardstruc.h` in the `examples\postcard` subdirectory. This header is an identical representation of the message in the message set in the Control Center.

When a user sends a postcard, the application puts a message to the queue associated with the *Postcard* message flow (MQSI_POSTCARD_INPUT_QUEUE). The message flow provides the following message processing:

- The MQInput node retrieves messages from the input queue MQSI_POSTCARD_INPUT_QUEUE.
- The input node passes the message to node **AddCountry**, an instance of the IBMPrimitive Compute node. This node enhances the content of the message by adding the Country field, containing the country of the location you selected when you sent the postcard (for example, if you selected “Adelaide”, it adds “Australia”).
- The **AddCountry** node now passes the message to the MQOutput node, which puts it to the output queue (MQSI_POSTCARD_OUTPUT_QUEUE).
- The receiving postcard application retrieves the message from the output queue, reads and interprets the content, and displays the element content in the corresponding fields of the Postcard user interface window.

Although this application has a message set already defined (see “Importing and deploying the MQSeries Integrator resources on Windows NT” on page 54), you can create a message set based on a C header file (like `postcardstruc.h`) by importing that C structure into the message repository using the import function of the Control Center. This is explained in detail in *MQSeries Integrator Using the Control Center*.

Note: Do not try this after importing PostcardMS; this would create duplicate entries and cause problems with deployment.

Building and using a message flow

This verification scenario illustrates how you define a basic message flow, how you assign the resources to the broker, and deploy your changes. It uses MQSeries Explorer to send messages through the message flow you create. It does not use any defined message sets. It assumes you are using the sample broker that you created in “Configuring your broker domain” on page 41. If

you want to use another broker, you must ensure that you create it (using **mqsicreatebroker**) and define it in the configuration repository (from the *Topology* view in the Control Center).

The tasks assume that the broker, sample queue managers, Control Center, listeners, sender channels, and Configuration Manager are running.

The following tasks are described:

- “Creating the MQSeries resources”.
- “Creating a message flow”.
- “Assigning the message flow to the broker” on page 64.
- “Testing the message flow” on page 65.

Creating the MQSeries resources

This verification test needs two queues, one for input, the other for output. To create the queues locally on Sun Solaris, type the following from a command line:

```
runmqsc MQSI_SAMPLE_QM
define qlocal(MQSI_INQ)
define qlocal(MQSI_OUTQ)
```

Creating a message flow

You must now create the message flow that will process the messages you put to your input queue using the Control Center on Windows NT. The message flow is very simple: it retrieves the message from the input queue and puts it to the output queue.

1. Select *File* → *New Workspace* to create a new (untitled) workspace. If you have run the verification applications, and have your sample workspace already open, you can create your new message flow in this workspace if you prefer.
2. Select the Message Flows tab and select the *Designer* view.
3. Right-click on the Message Flows root and select *Create* → *Message flow*.
4. Enter the name MQSI_TEST. Click **Finish**. The new message flow appears in the tree view.
5. Expand the IBMPrimitives tree to display the supplied nodes.
6. Select the MQSI_TEST message flow in the left-hand pane. Drag and drop an MQInput node into the right-hand pane.
7. Right-click the MQInput node in the right-hand pane and select *Properties*. On the Basic tab, type the MQSeries input queue name of your input queue (MQSI_INQ). Click **OK**.
8. Drag and drop an MQOutput node into the right-hand pane.

Building and using a message flow

9. Right-click the MQOutput node in the right-hand pane and select *Properties*. On the Basic tab, type in the queue manager name (MQSI_SAMPLE_QM) and the queue name (MQSI_OUTQ) for the output queue. Click **OK**.
10. Right-click the MQInput node and select *Connect* → *Out*. This gives you a connector attached to your mouse. Drag this to the MQOutput node and drop by left-clicking. The connector attaches itself to the input terminal.
11. You have now completed your first message flow. Select *File* → *Check in* → *All (Save to Shared)*. This checks in all the resources to the configuration repository.

Assigning the message flow to the broker

Now you have created a message flow, you have to tell MQSeries Integrator where you want to run that message flow (that is, on which broker). To do this, you must assign the message flow to your broker.

1. Select the *Assignments* view.
2. Expand the broker name (MQSI_SAMPLE_BROKER) to display the broker's execution groups. The sample broker currently just has one execution group, the default one (called *default*) which is always created whenever you create a broker using `mqsicreatebroker`.
3. Right-click the default execution group and select *Check Out*. This locks the execution group for you.
4. Expand the *Message flows* tree in the center pane. This displays all the message flows available for assignment.
5. Find MQSI_TEST and drag and drop it on the default execution group in the right-hand pane, where you can see the graphic of the broker and the default execution group. You can drop a message flow only on an execution group (not on the broker itself).
6. Check in the execution group by right-clicking the default execution group in the left-hand pane and selecting *Check In*.

Deploying the message flow to the broker

Assignment makes the connection between a message flow and a broker, but it is only when you deploy the change that the Configuration Manager updates the broker with the configuration stored in the configuration repository.

1. Before you can deploy any changes, you must have checked in everything that you have updated. If you have followed the instructions in this section, all the relevant resources are checked in. However, if you are in any doubt, you can check everything in by selecting *File* → *Check in* → *All (Save to Shared)*.
2. In the *Assignments* view, right-click the broker name in the left-hand pane.

3. Select *Deploy* → *Complete Assignments Configuration*. When the Configuration Manager receives this request from the Control Center, it sends messages to the broker to give it the updated information it needs to be able to support your new message flow.
4. Check the deploy by changing to the *Log* view and clicking the refresh button (the green icon above the log pane). Check for success messages. There might be a slight time delay before the messages appear.
5. View the deployed configuration graphically in the *Operations* view. Refresh this view and the broker, execution group and message flow are displayed with green lights, to show they are all active.

Testing the message flow

Note: SupportPac IH03 contains a GUI-based utility that can be used to test message flows. It allows users to put files containing, for example, XML messages into the message flow.

To test your message flow, follow the steps below:

1. Change to the directory containing the MQSeries sample programs:
`/opt/mqm/samp/bin`
2. Enter the following command:
`amqspout MQSI_INQ MQSI_SAMPLE_QM`
3. Press **Enter**.
4. Enter the text of your message.
5. Press **Enter** twice to quit the program.
6. Use either `amqsget` or `amqsbcbg` to get the message you entered. For example:
`amqsget MQSI_OUTQ MQSI_SAMPLE_QM`
7. Your message is now displayed.

Your test is complete.

Appendix A. System changes after installation

This appendix describes the changes that installation and configuration have made on the systems you have set up in your broker domain. It assumes that you have followed the guidance and details of configuration given in “Chapter 4. Configuring a broker domain” on page 29.

This appendix contains these sections:

- “Directory structure”.
- “Environment variables” on page 68.
- “Database contents” on page 69.
- “Default MQSeries resources” on page 71.

The following are the default home directories:

- MQSeries Integrator

/opt/mqsi

- DB2

/opt/IBMDB2

Written as <db2_root> wherever it appears in this chapter.

Directory structure

The following tables list the subdirectories created and populated within your home directories when you install the product. It also provides a brief description of the contents.

All files are installed with default security: all users can access and execute these files. You can use standard operating system facilities to impose stricter security on these files, or a subset of files, if you choose.

Table 4. /opt/mqsi/ directory structure after installation

| Directory Names | Contents |
|-----------------|--------------------------|
| bin | Executable commands |
| classes | Java class files |
| CmdAsst | Command Assistant files |
| docs | PDF files |
| icu16 | icu data files |
| include | Header files for samples |
| jre | Java Runtime Environment |

Directory structure

Table 4. /opt/mqsi/ directory structure after installation (continued)

| Directory Names | Contents |
|-----------------|---|
| lib | Shared library files |
| lil | MQSeries Integrator lils |
| merant | Merant Drivers |
| messages | Description files for messages and exceptions |
| readmes | Readme files |
| sample | Samples |
| template | Template files |
| tivready | Tivoli files |

Table 5. /var/mqsi directory structure after installation

| Directory | Contents |
|-----------|------------------------------------|
| brokers | Brokers directory |
| locks | Named mutex files |
| log | MQSeries Integrator logs directory |
| messages | Messages directory |
| odbc | ODBC files |
| registry | Registry files |
| users | Users directory |

Environment variables

The following environment variables are required to contain specific values for MQSeries Integrator:

- CLASSPATH
- LANG
- LC_ALL
- LD_LIBRARY_PATH
- MQSI_REGISTRY
- NLSPATH
- ODBCINI
- PATH

The following environment variables may be required to contain specific values if you use NEONRules and NEONFormatter Support:

- ICU_DATA

- MQSI_PARAMETERS_FILE
- MQSI_PRELOAD
- NEON_CATALOGUES
- NEON_ROOT
- NN_CONFIG_FILE_PATH

To ensure you get the latest environment variables you must refer to the `readme.txt` on the main product CD.

A profile that contains some of the required environment variables is shipped with this product. You must run this profile or a copy of it:

```
/opt/mqsi/sample/profiles/profile.sol
```

For more information on NEON environment variables see *MQSeries Integrator Using the Control Center*.

Database contents

When you create MQSeries Integrator resources following installation, database tables are created for your broker within the database you specify in the create command. The database you specify can be created for each set of tables or it can be shared. The database tables schema inherits its name from the data source specified in the create command.

Table 6 shows the tables that are created by the `mqsicreatebroker` command in the broker database. The tables are created when you create the first broker. When you create further brokers specifying the same database, new rows are created for each broker. Every row created in the table includes the broker name; therefore each row is unique to a single broker.

You must add all these database tables into your standard backup and recovery routines to ensure you can recover from system failures and other emergencies.

Table 6. Database tables for brokers

| Table name | Description |
|--------------------|--|
| BACLENTRIES | ACL entries |
| BCLIENTUSER | Maps client identifiers to durable subscriptions |
| BGROUPNAME | Publish/subscribe principals: groups |
| BLOGICALTOPHYSNAME | Maps logical to physical names |
| BMQEPUBDEST | Contains list of MQSeries Everyplace queues on which messages can be placed. |

Database contents

Table 6. Database tables for brokers (continued)

| Table name | Description |
|------------------|---|
| BMQEPUBMSGIN | Holds persistent MQSeries Everyplace publish/subscribe messages waiting to be processed by the broker. |
| BMQEPUBMSGOUT | Holds persistent MQSeries Everyplace publish/subscribe messages to be placed on an MQSeries Everyplace queue. |
| BMQESTDMSGIN | Holds persistent MQSeries Everyplace non-publish/subscribe messages waiting to be processed by the broker. |
| BMQESTDMSGOUT | Holds persistent messages to be sent to MQSeries Everyplace destinations. |
| BMQPSTOPOLOGY | Publish/subscribe neighbor information |
| BNBRCONNECTIONS | Inter-broker neighbor connection information |
| BPHYSICALFILE | Physical file mapping |
| BPUBLISHERS | Registered publishers |
| BRETAINEDPUBS | Retained publications |
| BRMCONFIG | Broker configuration details |
| BROKERA | Broker process details to support recovery |
| BROKERAEG | Execution group details to support recovery |
| BROKERRESOURCES | Broker resources |
| BSCADADEST | Contains list of SCADA clients to which messages can be sent. |
| BSCADAMSGIN | Holds persistent SCADA messages waiting to be processed by the broker. |
| BSCADAMSGOUT | Holds persistent messages to be sent to SCADA destinations. |
| BSUBSCRIPTIONS | Durable subscription information |
| BTOPOLOGY | Inter-broker neighbor information |
| BUSERCONTEXT | Maps client identifiers to context information |
| BUSERMEMBERSHIP | Publish/subscribe principals: membership |
| BUSERNAME | Publish/subscribe principals: users |
| BWFFRELATIONSHIP | Workfile details |

These tables are maintained by processes that are internal to MQSeries Integrator components. You must not access these tables by any other means, or change the access authority required by MQSeries Integrator.

Default MQSeries resources

When you create MQSeries Integrator components, some MQSeries resources are created for their use. Table 7 lists all these MQSeries resources, and indicates the component associated with the queue manager on which they are created. For details of which resources are created by which create commands, see the command descriptions in the *MQSeries Integrator Administration Guide*.

All resource names start with the reserved characters “SYSTEM”. Therefore, you should not find any conflict of names. There is one exception if you are using MQSeries Publish/Subscribe. It defines the queue SYSTEM.BROKER.CONTROL.QUEUE, which is also used by MQSeries Integrator. However, the use is compatible and you do not have to take any action to continue using this queue.

Table 7. MQSeries Integrator default objects

| Resource name | Type | Queue manager | Description |
|------------------------------------|-------|------------------|--|
| SYSTEM.BROKER.ADMIN.QUEUE | queue | broker | Target for messages sent by the Configuration Manager and commands to modify the broker's configuration and operation. |
| SYSTEM.BROKER.CONTROL.QUEUE | queue | broker | Target for publish/subscribe control requests from applications. A queue of this exact name is used by the MQSeries Publish/Subscribe. You might therefore already have defined a queue of this name on the queue manager. You can continue to use this same queue as you migrate to MQSeries Integrator. |
| SYSTEM.BROKER.EXECUTIONGROUP.QUEUE | queue | broker | Target for messages to the broker. |
| SYSTEM.BROKER.EXECUTIONGROUP.REPLY | queue | broker | Target for response messages for the broker from the User Name Server. |
| SYSTEM.BROKER.INTERBROKER.QUEUE | queue | broker | Target for publications from neighbor brokers. |
| SYSTEM.BROKER.MODEL.QUEUE | queue | all | Model for dynamic response queues. |
| SYSTEM.BROKER.SECURITY.QUEUE | queue | User Name Server | Target for request messages to the User Name Server. queue is used by brokers, the Configuration Manager, and the command line tools. |

Note: These resources are defined in addition to the MQSeries product default objects, which are defined when the MQSeries Messaging product is installed. You can find a full description of these default objects in *MQSeries System Administration*.

MQSeries resources

Appendix B. Setting up an Oracle8 broker database on MQSeries Integrator

This chapter describes how to set up an Oracle8 broker database on MQSeries Integrator and is intended for an Oracle database administrator (DBA).

You are recommended to create a new database instance. Although it is possible to use an existing database instance, you are **not** recommended to do this.

These instructions assume that:

- The database instance will be created following Oracle documentation.
- Oracle communications will be set up using SQL*Net following Oracle documentation.
- Oracle operations are conducted within the correct Oracle environment (for example ORACLE_HOME).

Naming

MQSeries Integrator uses the Data Source Name (DSN) defined in the ODBC setup. You do not have to use a particular named instance for use as a broker database.

Schema

MQSeries Integrator does not demand a particular schema or set of tablespaces for keeping broker information. When you create a broker, the tables are created in the database with an ownership defined by the user identifier specified on the command line (the user ID must already be known to the database). For example:

```
mqsicreatebroker BRK -i bid -a bpw -q QM -n BDB -u dbid -p dbpw
```

creates tables all owned by dbid (like DBID.BROKERA). The tablespace used to hold these tables is the default tablespace for the Oracle user ID specified (the default is normally SYSTEM).

Sizing

When you create a broker, MQSeries Integrator does not generate much data to be stored in the database. Deployment of a complex flow to the broker can consume more database space but still not a large amount. If you create an instance specifically for use as a broker database, taking the default settings

Sizing

defined by the 'dbassist' tool should be sufficient for most applications. 50 MB is enough for a custom setup. If a tablespace is set up specifically for use by the brokers, this can be extended at a later date.

User Identifier

The Oracle user ID that you use to store broker information needs the privileges: connect, resource, and create table. For example, the following is sufficient:

```
CREATE USER dbid IDENTIFIED BY dbpw; GRANT CONNECT TO dbid; GRANT RESOURCE TO dbid; GRANT CREATE TABLE TO dbid;
```

and optionally:

```
ALTER USER dbid DEFAULT TABLESPACE brktspace;
```

This user ID can be the same as the operating system ID that will be used to create the broker or it can be specific to the database. If it is specific to the database, use the '-u' and '-p' flags when you create the broker.

ODBC

1. When you have created and started the database, and SQL*Net has been configured (using listener.ora and tnsnames.ora) and started to check that the database is accessible through the SQL*Net interface. First check that the SQL*Net listener is running by using:

```
lsnrctl status
```

or checking the process list for tnslnsr using:

```
ps -ef | grep tnslnsr
```

2. Oracle provides a utility tnsping to check that a configured DSN (Oracle Global Database Name) is accessible. For example:

```
tnsping myDSN
```

However, it is useful to check further and test the user ID using SQL*Plus. For example:

```
sqlplus dbid/dbpw@myDSN
```

3. This connects you to the SQL*Plus application and shows in a process list that the connection is not local (that is, the connection is using SQL*Net). Use:

```
ps -ef | grep myDSN
```

One of the entries in the list should be something like:

```
myid 1234 5678  
0 10:55:57 ? 0:00 ora1emyDSN (LOCAL=N0)
```


4. Update the MQSeries Integrator ODBC description file `/var/mqsi/odbc/.odbc.ini` to add entries for your broker database. At the head of the file, in the [ODBC Data Sources] section, add an entry specifying the DSN that the broker will use. This might be different from the DSN defined by SQL*Net if required. For example:

```
[ODBC Data Sources] MYBRKDSN=MERANT
3.60 Oracle 8 driver
```

5. Create a stanza to define the driver path and Oracle DSN for your broker DSN, for example:

```
[MYBRKDSN] Driver=/opt/mqsi/merant/lib/UKor815.so
Description=Oracle8
ServerName=myDSN
EnableDescribeParam=1
OptimizePrepare=1
```

6. You should now be able to create a broker with a command like:

```
mqsicreatebroker BRK -i uid -a pwd -q QM -n MYBRKDSN -u dbid -p dbpw
```

Appendix C. Uninstalling MQSeries Integrator

This chapter gives details of the processes that allow you to uninstall any one or all of the MQSeries Integrator components on Sun Solaris.

Before you start

Before you start to uninstall MQSeries Integrator for Sun Solaris you must:

1. Log on as root and use the `mqsilist` command to check what components are defined on the machine
2. Stop any brokers that are running
3. Stop your User Name Server (if you have one)

Uninstalling

To uninstall on Sun Solaris, use the `pkgrm` command. For example, to remove MQSeries Integrator, use the following command:

```
pkgrm mqs i
```

To uninstall a CSD level, use the command:

```
pkgrm mqs i -updn
```

where *n* is the update level.

When uninstalling, the contents of the broker database tables are deleted, not the tables themselves.

Contacting your IBM Support Center

If you are unable to resolve problems that you find when you use MQSeries Integrator, or if you are directed to do so by an error message generated by MQSeries Integrator, you can request assistance from your IBM support center.

Before you contact them, use the checklist below to gather key information. Some items may not necessarily be relevant in every situation. But you should provide as much information as possible to enable the IBM support center to recreate your problem.

- For MQSeries Integrator:
 - CSDs applied.
 - E-fixes applied.

- All current trace and error logs, including relevant Sun Solaris platform syslog or Windows NT Event log entries. User trace log files at debug level should be obtained for all relevant message flows and should preferably be formatted.
- A list of the components installed. This should include details of the number of machines and their operating systems, the number of brokers and the machine on which they are running, and the existence and details of any User Name Servers.
- The file obtained by exporting the message flow that was causing the problem, if known.
- The file obtained by exporting your workspace. This action is performed from the Control Center; see the *MQSeries Integrator Using the Control Center* manual for details of how to do this.
- The files obtained by exporting all relevant message sets. This action is performed for each message set by using the `mqsiimpexpmsgset` command with the `-e` flag set.
- A sample of the messages being used when the problem arose.
- If relevant, the report file from the C or COBOL importer. This is located in the directory from which the file import was attempted.
- For MQSeries:
 - CSDs applied.
 - E-fixes applied.
 - All current trace and error logs, including relevant Sun Solaris platform syslog or Windows NT Event log entries and First Failure Support Technology™ (FFST) output files. You can find these files, which have the extension FDC, in the errors subdirectory within the MQSeries home directory.
 - Details of MQSeries client software, if appropriate.
- For each database you are using:
 - Product and release level (for example, DB2 7.1).
 - CSDs applied.
 - E-fixes applied.
 - All current trace and error logs, including relevant Sun Solaris platform syslog or Windows NT Event log entries and FFST™ output files. Check database product documentation for where to find these files.
- For Sun Solaris:
 - Version. You can find the version of Sun Solaris installed by using the `uname -a` command.
 - Service level applied.
- For Windows NT:
 - Version.

- Service Pack level.
- The version of the system files `msvcrt.dll`, `msvcp60.dll`, `msvcirt.dll`, and `mfc42.dll`. You can find these files in the `WINNT\system32` directory. Use the Windows NT Explorer file properties to display the versions.
- Details of the operation you were performing, the results that occurred, and the results you were expecting.

Appendix D. Applying maintenance

Maintenance updates are supplied on CD in the form of a Program Temporary Fix (PTF), referred to as a Corrective Service Diskette (CSD). You can find the latest information about available CSDs on the Internet, at the address given in "MQSeries information available on the Internet" on page 98.

You can also download CSDs from this Web site.

Applying maintenance to MQSeries Integrator

Before applying any maintenance, read the file `memo.ptf` and any `readme.txt` files in the root directory of the CD. If you need to apply a CSD:

1. Stop all brokers that are running.
2. Change to the directory containing the CSD.
3. Type the following command:

```
pkgadd -d .
```

4. Select:

```
mqsi-updn
```

Where *n* is the level of update.

To see the current level of corrective service, enter the command:

```
pkginfo | grep mqsi
```

After installation of the CSD, you can restart your broker or brokers.

The current level is the highest value of *n* in the displayed values of `mqsi-updn`.

To remove a CSD, enter the command:

```
pkgrm mqsi-updn
```

Where *n* is the number of the CSD to remove.

Note: CSDs are cumulative, therefore you do not need to apply CSD1 before you can apply CSD2. When you have installed a CSD, you are prevented from installing a previous CSD without first restoring the system using the backed-up files.

Restoring a service level

Restoring a previous service level

Because CSDs are cumulative, you must uninstall the most recent CSD you applied before attempting to uninstall any previous CSDs. For example, if you have installed CSD1, CSD2, and CSD3 on your machine and you want to revert to the CSD1 level of code, you must first uninstall CSD3 and then uninstall CSD2.

Applying maintenance to IBM DB2 Universal Database

You can also obtain information about the current status of maintenance of this product, and download fix packs for DB2 from the Web site identified in “MQSeries information available on the Internet” on page 98.

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

Notices

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.

Trademarks

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

| | |
|------------|----------------------------------|
| AIX | AS/400 |
| DB2 | DB2 Universal Database |
| FFST | First Failure Support Technology |
| IBM | IBMLink |
| MQSeries | OS/390 |
| SupportPac | VSE/ESA |

Tivoli is a trademark of Tivoli Systems Inc. in the United States, other countries, or both.

Windows and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, JDBC, and JDK are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Glossary of terms and abbreviations

This glossary defines MQSeries Integrator terms and abbreviations used in this book. If you do not find the term you are looking for, see the index or the *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

This glossary includes terms and definitions from the *American National Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute. Copies may be ordered from the American National Standards Institute, 11 West 42 Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.

A

Access Control List (ACL). The list of principals that have explicit permissions (to publish, to subscribe to, and to request persistent delivery of a publication message) against a topic in the topic tree. The ACLs define the implementation of topic-based security.

ACL. Access Control List.

AMI. Application Messaging Interface.

Application Messaging Interface (AMI). The programming interface provided by MQSeries that defines a high level interface to message queuing services. See also *MQI* and *JMS*.

B

blob. Binary Large Object. A block of bytes of data (for example, the body of a message) that has no discernible meaning, but is treated as one solid entity that cannot be interpreted. Also written as BLOB.

broker. See *message broker*.

broker domain. A collection of brokers that share a common configuration, together with the single Configuration Manager that controls them.

C

callback function. See *implementation function*.

category. An optional grouping of messages that are related in some way. For example, messages that relate to a particular application.

check in. The Control Center action that stores a new or updated resource in the configuration or message repository.

check out. The Control Center action that extracts and locks a resource from the configuration or message repository for local modification by a user. Resources from the two repositories can only be worked on when they are checked out by an authorized user, but can be viewed (read only) without being checked out.

collective. A hyperconnected (totally connected) set of brokers forming part of a multi-broker network for publish/subscribe applications.

configuration. In the broker domain, the brokers, execution groups, message flows and message sets assigned to them, topics and access control specifications.

Configuration Manager. A component of MQSeries Integrator that acts as the interface between the configuration repository and an executing set of brokers. It provides brokers with their initial configuration, and updates them with any subsequent changes. It maintains the broker domain configuration.

configuration repository. Persistent storage for broker configuration and topology definition.

connector. See *message processing node connector*.

Glossary

content-based filter. An expression that is applied to the content of a message to determine how the message is to be processed.

context tag. A tag that is applied to an element within a message to enable that element to be treated differently in different contexts. For example, an element could be mandatory in one context and optional in another.

Control Center. The graphical interface that provides facilities for defining, configuring, deploying, and monitoring resources of the MQSeries Integrator network.

D

datagram. The simplest form of message that MQSeries supports. Also known as *send-and-forget*. This type of message does not require a reply. Compare with *request/reply*.

debugger. A facility on the *Message Flows* view in the Control Center that enables message flows to be debugged.

deploy. Make operational the configuration and topology of the broker domain.

destination list. A list of internal and external destinations to which a message is sent. These can be nodes within a message flow (for example, when using the *RouteToLabel* and *Label* nodes) or MQSeries queues (when the list is examined by an *MQOutput* node to determine the final target for the message).

distribution list. A list of MQSeries queues to which a message can be put using a single statement.

Document Type Definition (DTD). The rules that specify the structure for a particular class of SGML or XML documents. The DTD defines the structure with elements, attributes, and notations, and it establishes constraints for how each element, attribute, and notation can be used within the particular class of documents. A DTD is analogous to a database schema in that the DTD completely describes the structure for a particular markup language.

DTD. Document Type Definition

E

e-business. A term describing the commercial use of the Internet and World Wide Web to conduct business (short for electronic-business).

element. A unit of data within a message that has business meaning, for example, street name

element qualifier. See *context tag*.

ESQL. Extended SQL. A specialized set of SQL statements based on regular SQL, but extended with statements that provide specialized functions unique to MQSeries Integrator.

exception list. A list of exceptions that have been generated during the processing of a message, with supporting information.

execution group. A named grouping of message flows that have been assigned to a broker. The broker is guaranteed to enforce some degree of isolation between message flows in distinct execution groups by ensuring that they execute in separate address spaces, or as unique processes.

Extensible Markup Language (XML). A W3C standard for the representation of data.

external reference. A reference within a message set to a component that has been defined outside the current message set. For example, an integer that defines the length of a string element might be defined in one message set but used in several message sets.

F

field reference. A sequence of period-separated values that identify a specific field (which might be a structure) within a message tree. An example of a field reference might be something like *Body.Invoice.InvoiceNo*.

filter. An expression that is applied to the content of a message to determine how the message is to be processed.

format. A format defines the internal structure of a message, in terms of the fields and order of those fields. A format can be self-defining, in which case the message is interpreted dynamically when read.

G

graphical user interface (GUI). An interface to a software product that is graphical rather than textual. It refers to window-based operational characteristics.

I

implementation function. Function written by a third-party developer for a plug-in node or parser. Also known as a *callback function*.

input node. A message flow node that represents a source of messages for the message flow.

installation mode. The installation mode can be Full, Custom, or Broker only. The mode defines the components of the product installed by the installation process on Windows NT systems.

J

Java Database Connectivity (JDBC). An application programming interface that has the same characteristics as **ODBC** but is specifically designed for use by Java database applications.

Java Development Kit (JDK). A software package that can be used to write, compile, debug, and run Java applets and applications.

Java Message Service (JMS). An application programming interface that provides Java language functions for handling messages.

Java Runtime Environment (JRE). A subset of the Java Development Kit (JDK) that contains the core executables and files that constitute the standard Java platform. The JRE includes the Java Virtual Machine, core classes and supporting files.

JDBC™. Java Database Connectivity.

JDK™. Java Development Kit.

JMS. Java Message Service. See also *AMI* and *MQI*.

JRE. Java Runtime Environment.

L

local error log. A generic term that refers to the logs to which MQSeries Integrator writes records on the local system. On Windows NT, this is the Event log. On UNIX systems, this is the syslog. See also *system log*. Note that MQSeries records many events in the log that are not errors, but information about events that occur during operation, for example, successful deployment of a configuration.

M

message broker. A set of execution processes hosting one or more message flows.

messages. Entities exchanged between a broker and its clients.

message dictionary. A repository for (predefined) message type specifications.

message domain. The value that determines how the message is interpreted (parsed). The following domains are recognized:

- MRM, which identifies messages defined using the Control Center
- NEONMSG², which identifies messages created using the NEONFORMATTER user interfaces.
- XML, which identifies messages that are self-defining
- BLOB, which identifies messages that are undefined

2. The message domain NEON is also recognized for compatibility with previous releases.

Glossary

You can also create your own message domains: if you do so, you must supply your own message parser.

message flow. A directed graph that represents the set of activities performed on a message or event as it passes through a broker. A message flow consists of a set of message processing nodes and message processing node connectors.

message flow component. See *message flow*.

message parser. A program that interprets a message bitstream.

message processing node. A node in the message flow, representing a well defined processing stage. A message processing node can be one of several primitive types or can represent a subflow.

message processing node connector. An entity that connects the output terminal of one message processing node to the input terminal of another. A message processing node connector represents the flow of control and data between two message flow nodes.

message queue interface (MQI). The programming interface provided by MQSeries queue managers. The programming interface allows application programs to access message queuing services. See also *AMI* and *JMS*.

message repository. A database holding message template definitions.

message repository manager (MRM). A component of the Configuration Manager that handles message definition and control. A message defined to the MRM has a message domain set to MRM.

message set. A grouping of related messages.

message template. A named and managed entity that represents the format of a particular message. Message templates represent a business asset of an organization.

message type. The logical structure of the data within a message. For example, the number and location of character strings.

metadata. Data that describes the characteristic of stored data.

MQI. Message queue interface.

MQIsdp. MQSeries Integrator SCADA device protocol. A lightweight publish/subscribe protocol flowing over TCP/IP.

MQRFH. An architected message header that is used to provide metadata for the processing of a message. This header is supported by MQSeries Publish/Subscribe.

MQRFH2. An extended version of MQRFH, providing enhanced function in message processing.

MQSeries Everyplace. A generally available MQSeries product that provides proven MQSeries reliability and security in a mobile environment.

MRM. Message Repository Manager.

multilevel wildcard. A wildcard that can be specified in subscriptions to match any number of levels in a topic.

N

node. See *message processing node*.

O

ODBC. Open Database Connectivity.

Open Database Connectivity. A standard application programming interface (API) for accessing data in both relational and non-relational database management systems. Using this API, database applications can access data stored in database management systems on a variety of computers even if each database management system uses a different data storage format and programming interface. ODBC is

based on the call level interface (CLI) specification of the X/Open SQL Access Group.

output node. A message processing node that represents a point at which messages flow out of the message flow.

P

plug-in. An extension to the broker, written by a third-party developer, to provide a new message processing node or message parser in addition to those supplied with the product. See also *implementation function* and *utility function*.

point-to-point. Style of messaging application in which the sending application knows the destination of the message. Compare with *publish/subscribe*.

POSIX. Portable Operating System Interface For Computer Environments. An IEEE standard for computer operating systems (for example, AIX[®] and Sun Solaris).

predefined message. A message with a structure that is defined before the message is created or referenced. Compare with *self-defining message*.

primitive. A message processing node that is supplied with the product.

principal. An individual user ID (for example, a log-in ID) or a group. A group can contain individual user IDs and other groups, to the level of nesting supported by the underlying facility.

property. One of a set of characteristics that define the values and behaviors of objects in the Control Center. For example, message processing nodes and deployed message flows have properties.

publication node. An end point of a specific path through a message flow to which a client application subscribes. A publication node has an attribute, subscription point. If this is not specified, the publication node represents the default subscription point for the message flow.

publish/subscribe. Style of messaging application in which the providers of information (publishers) are decoupled from the consumers of that information (subscribers) using a broker. Compare with *point-to-point*. See also *topic*.

publisher. An application that makes information about a specified topic available to a broker in a publish/subscribe system.

Q

queue. An MQSeries object. Message queuing applications can put messages on, and get messages from, a queue. A queue is owned and maintained by a queue manager. Local queues can contain a list of messages waiting to be processed. Queues of other types cannot contain messages: they point to other queues, or can be used as models for dynamic queues.

queue manager. A system program that provides queuing services to applications. It provides an application programming interface (the MQI) so that programs can access messages on the queues that the queue manager owns.

R

retained publication. A published message that is kept at the broker for propagation to clients that subscribe at some point in the future.

request/reply. Type of messaging application in which a request message is used to request a reply from another application. Compare with *datagram*.

rule. A rule is a definition of a process, or set of processes, applied to a message on receipt by the broker. Rules are defined on a message format basis, so any message of a particular format will be subjected to the same set of rules.

S

SCADA. Supervisory, Control, And Data Acquisition.

Glossary

self-defining message. A message that defines its structure within its content. For example, a message coded in XML is self-defining. Compare with *pre-defined message*.

send and forget. See *datagram*.

setup type. The definition of the type of installation requested on Windows NT systems. This can be one of **Full**, **Broker only**, or **Custom**.

shared. All configuration data that is shared by users of the Control Center. This data is not operational until it has been deployed.

signature. The definition of the external characteristics of a message processing node.

single-level wildcard. A wildcard that can be specified in subscriptions to match a single level in a topic.

stream. A method of topic partitioning used by MQSeries Publish/Subscribe applications.

subscriber. An application that requests information about a specified topic from a publish/subscribe broker.

subscription. Information held within a publication node, that records the details of a subscriber application, including the identity of the queue on which that subscriber wants to receive relevant publications.

subscription filter. A predicate that specifies a subset of messages to be delivered to a particular subscriber.

subscription point. An attribute of a publication node that differentiates it from other publication nodes on the same message flow and therefore represents a specific path through the message flow. An unnamed publication node (that is, one without a specific subscription point) is known as the default publication node.

Supervisory, Control, And Data Acquisition. A broad term, used to describe any form of remote telemetry system used for gathering data from remote sensor devices (for example, flow rate

meters on an oil pipeline) and for the near real time control of remote equipment (for example, pipeline valves).

system log. A generic term used in the MQSeries Integrator messages (BIPxxx) that refers to the local error logs to which records are written on the local system. On Windows NT, this is the Event log. On UNIX systems, this is the syslog. See also *local error log*.

T

terminal. The point at which one node in a message flow is connected to another node. Terminals enable you to control the route that a message takes, depending whether the operation performed by a node on that message is successful.

topic. A character string that describes the nature of the data that is being published in a publish/subscribe system.

topic based subscription. A subscription specified by a subscribing application that includes a topic for filtering of publications.

topic security. The use of ACLs applied to one or more topics to control subscriber access to published messages.

topology. In the broker domain, the brokers, collectives, and connections between them.

transform. A defined way in which a message of one format is converted into one or more messages of another format.

U

Uniform Resource Identifier. The generic set of all names and addresses that refer to World Wide Web resources.

Uniform Resource Locator. A specific form of URI that identifies the address of an item on the World Wide Web. It includes the protocol followed by the fully qualified domain name (sometimes called the host name) and the request. The Web server typically maps the

request portion of the URL to a path and file name. Also known as Universal Resource Locator.

URI. Uniform Resource Identifier

URL. Uniform Resource Locator

User Name Server. The MQSeries Integrator component that interfaces with operating system facilities to determine valid users and groups.

utility function. Function provided by MQSeries Integrator for the benefit of third-party developers writing plug-in nodes or parsers.

W

warehouse. A persistent, historical datastore for events (or messages). The **Warehouse** node within a message flow supports the recording of information in a database for subsequent retrieval and processing by other applications.

wildcard. A character that can be specified in subscriptions to match a range of topics. See also *multilevel wildcard* and *single-level wildcard*.

wire format. This describes the physical representation of a message within the bit-stream.

W3C. World Wide Web Consortium. An international industry consortium set up to develop common protocols to promote evolution and interoperability of the World Wide Web.

X

XML. Extensible Markup Language.

Glossary

Bibliography

This section describes the documentation available for all current MQSeries Integrator products.

MQSeries Integrator Version 2.0.2 cross-platform publications

The MQSeries Integrator cross-platform publications are:

- *MQSeries Integrator Introduction and Planning*, GC34-5599
- *MQSeries Integrator Using the Control Center*, GC34-5602
- *MQSeries Integrator Messages*, GC34-5601
- *MQSeries Integrator Programming Guide*, SC34-5603
- *MQSeries Integrator Administration Guide*, SC34-5792
- *MQSeries Integrator ESQL Reference*, SC34-5923

These books are all available in hardcopy.

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.

MQSeries Integrator Version 2.0.2 platform-specific publications

Each MQSeries Integrator product provides one platform-specific installation guide, which is supplied in hardcopy.

MQSeries Integrator for AIX Version 2.0.2

MQSeries Integrator for AIX Installation Guide, GC34-5841

MQSeries Integrator for HP-UX Version 2.0.2

MQSeries Integrator for HP-UX Installation Guide, GC34-5907

MQSeries Integrator for Sun Solaris Version 2.0.2

MQSeries Integrator for Sun Solaris Installation Guide, GC34-5842

MQSeries Integrator for Windows NT Version 2.0.2

MQSeries Integrator for Windows NT Installation Guide, GC34-5600

MQSeries Everyplace publications

If you intend to connect MQSeries Everyplace applications to message flows that include the MQSeries Everyplace message flow nodes, you will find the following publications useful:

- *MQSeries Everyplace for Multiplatforms Version 1.1 Introduction*, GC34-5843
- *MQSeries Everyplace for Multiplatforms Version 1.1 Programming Guide*, SC34-5845
- *MQSeries Everyplace for Multiplatforms Version 1.1 Programming Reference*, SC34-5846

Bibliography

- *MQSeries Everyplace for Multiplatforms Version 1.1 Native Client Information*, SC34-5880

You can find these books on the MQSeries Web site (see “MQSeries information available on the Internet” on page 98). Translated versions of these books are also available in some languages from the same Web site.

NEONRules and NEONFormatter Support for MQSeries Integrator publications

The following publications are supplied on the product CD in PDF format, and are installed with the Documentation component.

- *NEONRules and NEONFormatter Support for MQSeries Integrator User's Guide*
- *NEONRules and NEONFormatter Support for MQSeries Integrator System Management Guide*
- *NEONRules and NEONFormatter Support for MQSeries Integrator Programming Reference for NEONRules*
- *NEONRules and NEONFormatter Support for MQSeries Integrator Programming Reference for NEONFormatter*
- *NEONRules and NEONFormatter Support for MQSeries Integrator Application Development Guide*

These books are provided in US English only.

Softcopy books

All the MQSeries Integrator books are available in softcopy formats.

Portable Document Format (PDF)

All books in the MQSeries Integrator library are supplied in US English only in a searchable PDF library on the product CD.

You can install the library as follows:

- On AIX, invoke `install -d` and select the documentation fileset. After installation, run the command `mqsidocs`. This launches Acrobat Reader and opens the PDF package.
- On HP-UX, invoke `swinstall -d` and select `MQSI-DOCS` from the menu. After installation, run the command `mqsidocs`. This launches Acrobat Reader and opens the PDF package.
- On Sun Solaris, invoke `pkgadd -d` and select `mqsi-docs` from the menu. After installation, run the command `mqsidocs`. This launches Acrobat Reader and opens the PDF package.
- On Windows NT, select the Online Documentation component on a custom installation, or do a full installation. After installation, select `Start—>Programs—>IBM MQSeries Integrator 2.0—>Documentation`.

In addition, PDF files for books that have been translated are installed into the location `mqsi_root/bin/book/pdf/<locale>` (on UNIX) or `mqsi_root\bin\book\pdf\<locale>` (on Windows NT) where `<locale>` is one of the following:

- **de_DE** for German
- **en_US** for US English
- **es_ES** for Spanish
- **fr_FR** for French
- **it_IT** for Italian
- **ja_JP** for Japanese
- **ko_KR** for Korean
- **pt_BR** for Brazilian Portuguese
- **zh_CN** for Simplified Chinese
- **zh_TW** for Traditional Chinese

An index file (in HTML format) that provides a link to each book is supplied for each language. For example, the French index file is called `indexfr.htm`. The files are stored in the following directory:

- On UNIX, `<mqsi_root>/docs/`
- On Windows NT, `<mqsi_root>\bin\book`

Each index file has an entry for every book: if a particular book has not been translated into the appropriate language for that index file, a link to the English PDF is included. You can use any Web browser to view the index file. On Windows NT, you can also access the index file through the *Start* menu.

The PDF file names for the English books are shown in Table 8.

Table 8. File names of MQSeries Integrator book PDFs

| Book title | File name |
|---|---------------------------|
| <i>MQSeries Integrator for AIX Installation Guide</i> | <code>bipaac04.pdf</code> |
| <i>MQSeries Integrator for HP-UX Installation Guide</i> | <code>bipcac00.pdf</code> |
| <i>MQSeries Integrator for Sun Solaris Installation Guide</i> | <code>bip7ac03.pdf</code> |
| <i>MQSeries Integrator for Windows NT Installation Guide</i> | <code>bipyac03.pdf</code> |
| <i>MQSeries Integrator Introduction and Planning</i> | <code>bipyab02.pdf</code> |
| <i>MQSeries Integrator Administration Guide</i> | <code>bipyag04.pdf</code> |
| <i>MQSeries Integrator Using the Control Center</i> | <code>bipyar03.pdf</code> |
| <i>MQSeries Integrator ESQL Reference</i> | <code>bipyae00.pdf</code> |
| <i>MQSeries Integrator Programming Guide</i> | <code>bipyal02.pdf</code> |
| <i>MQSeries Integrator Messages</i> | <code>bipyao02.pdf</code> |

The fifth character of the file name indicates the language of the book (**a** indicates US English). You can deduce the file names of translated books by using the following substitutions for the fifth character:

- **g** for German
- **s** for Spanish
- **f** for French
- **i** for Italian
- **j** for Japanese
- **k** for Korean
- **b** for Brazilian Portuguese
- **z** for Simplified Chinese
- **t** for Traditional Chinese

PDF files can be viewed and printed using the Adobe Acrobat Reader.

If you cut and paste examples of commands from PDF files to a command line for execution, you must check that the content is correct before you press Enter. Some characters might be corrupted by local system and font settings.

If you need to obtain the Adobe Acrobat Reader, or would like up-to-date information about the platforms on which the Acrobat Reader is supported, visit the Adobe Systems Inc. Web site at:

<http://www.adobe.com/>

PDF versions of all current MQSeries Integrator books are also available from the MQSeries product family Web site at:

<http://www.ibm.com/software/mqseries/>

MQSeries publications

The following books are referred to in this book to point you to the information you need to complete MQSeries Messaging product tasks as part of MQSeries Integrator tasks.

MQSeries publications

For Sun Solaris installation tasks you might need:

- *MQSeries for Sun Solaris Quick Beginnings*, GC33-1870.

This book is included, in hardcopy, in the MQSeries Integrator package.

It provides detailed planning and installation guidance.

For Windows NT installation tasks you might need:

- *MQSeries for Windows NT and Windows 2000 Quick Beginnings*, GC34-5389.

It provides detailed planning and installation guidance.

For planning and configuration tasks you might need:

- *MQSeries MQSC Command Reference*, SC33-1369.

This book contains the syntax of the MQSC commands.

- *MQSeries System Administration*, SC33-1873.

This book supports day-to-day management of local and remote MQSeries objects.

- *MQSeries Clients*, GC33-1632.

This book describes how to install, configure, use, and manage MQSeries clients.

- *MQSeries Intercommunication*, SC33-1872.

This book describes MQSeries Intercommunication between different platforms.

For a complete list of MQSeries product publications, refer to the information on the MQSeries Web site (given in “MQSeries information available on the Internet”).

MQSeries Publish/Subscribe publications

If you have installed MQSeries Publish/Subscribe, and plan to migrate to MQSeries Integrator Version 2, or to establish a mixed broker network, refer to:

- *MQSeries Publish/Subscribe User's Guide*, GC34-5269.

This book and the MQSeries Publish/Subscribe SDK package are available on the MQSeries Web site (given in “MQSeries information available on the Internet”).

DB2 publications

You may find the following publications useful when you set up and use DB2.

- *IBM DB2 Universal Database for Windows Quick Beginnings*, GC09-2971
- *IBM DB2 Universal Database Message Reference Volume 1*, GC09-2978
- *IBM DB2 Universal Database Message Reference Volume 2*, GC09-2979
- *IBM DB2 Universal Database Troubleshooting Guide*, GC09-2850

You can download these publications from the DB2 Web site at

<http://www.ibm.com/software/data/db2>

MQSeries information available on the Internet

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.
- | • Obtain information about complementary offerings by following these links:
 - | – IBM Business Partners
 - | – Partner Offerings (within *Related links*)
- Download an MQSeries SupportPac.

Index

A

Adobe Acrobat Reader 11
applying maintenance 81
assumptions, reviewing 34

B

broker 8
 connecting to Domain 50
 creating 45
 starting 49
broker domain
 assumptions 34
 checking configuration 55
 configuring 29, 41
 creating the broker 45
 creating the Configuration Manager 41
 database authorizations 40
 database setup 37
 deploying configuration 55
 MQSeries objects 71
 saving configuration 55
 simple configuration 29
 starting 46
 starting MQSeries listeners 47
 starting the broker 49
 starting the Configuration Manager 48
 starting the Control Center 49
 system updates 67

C

checking components 46
code page support 7
complementary offerings
 IBM Business Partners 99
 Partner Offerings 99
components, checking 46
configuration
 kernel 18
configuration components,
 installing 2
Configuration Manager 43
 creating 41
 starting 48
contacting IBM 77
Control Center
 initializing connection 49
 starting 49

D

database
 authorizations 40
 broker tables 46, 69
 connecting 37
 contents 69
 creating 37
 customizing 41
 initializing 38
 ODBC connection 39
 Oracle 73
 setting up on Sun Solaris 24
 tables 69
database installation
 DB2 5
database summary 7
DB2 8
 applying maintenance 82
 license agreement 5
 publications 98
default MQSeries resources 71
delivery media 15
directory structure 67
disk space required 4

E

environment variables
 CLASSPATH 68
 ICU_DATA 68
 LANG 68
 LC_ALL 68
 LD_LIBRARY_PATH 68
 MQSI_PARAMETERS_FILE 68
 MQSI_PRELOAD 68
 MQSI_REGISTRY 68
 NEON_CATALOGUES 68
 NEON_ROOT 68
 NLSPATH 68
 NN_CONFIG_FILE_PATH 68
 ODBCINI 68
 PATH 68

G

groups, setting up 36

H

hardware requirements 3

I

IBM Business Partners 99

information on the Internet
 complementary offerings 99
 MQSeries family libraries 99
 MQSeries products 98
 MQSeries SupportPacs 99
installation
 application programming 6
 before you start 16
 changes after 67
 configuration components 2
 connectivity 6
 delivery media 15
 disk space requirements 4
 error logs 27
 errors 27
 general hardware requirements 3
 installing runtime support 1
 kernel configuration 18
 LAN 25
 license agreement 7
 MQSeries 5
 MQSeries Integrator databases 6
 overview 1
 planning 3
 preparation 16
 prerequisites 19
 problems 27
 procedure 20
 runtime support 1
 security 12
 shared CD drive 25
 shared LAN drive 25
 silent 25
 software options 6
 software requirements 5
 verification 53
installing
 post-installation configuration and verification 2
installing on Sun Solaris 15

K

kernel configuration 18

L

LAN installation 25
license agreement
 DB2 5
 MQSeries Integrator 7

- M**
- maintenance
 - applying to DB2 82
 - applying to MQSeries Integrator 81
 - restoring previous level 82
- message flow
 - assigning to the broker 64
 - building and using 62
 - creating 63
 - creating MQSeries resources 63
 - deploying to the broker 64
 - testing 65
- message sets
 - import 54
- Migration 14
- MQSeries
 - publications 97
- MQSeries Everyplace
 - publications 95
- MQSeries Integrator on the Internet 98
- MQSeries Integrator
 - publications 95
 - national language 96
 - platform-specific 95
- MQSeries objects
 - broker 46
 - Configuration Manager 44
- MQSeries Publish/Subscribe
 - publications 98
- N**
- National Language Support 7
- NEON Interface 11
- NEONRules and NEONFormatter
 - Support
 - code page support restrictions 8
- NEONRules and NEONFormatter
 - Support publications 96
- O**
- ODBC 74
- ODBC connection
 - defining 39
- online documentation 11
- Oracle
 - database 73
 - ODBC 74
 - schema 73
- Oracle database
 - naming 73
 - ODBC 74
 - schema 73
 - setting up 73
- Oracle database (*continued*)
 - sizing 73
 - user ID 74
- P**
- Partner Offerings 99
- PDF (Portable Document Format) 96
- Portable Document Format (PDF) 96
- post-installation verification 2
- Postcard 60
 - how it works 61
 - running 60
- preparation
 - for installation 16
 - for uninstall 77
 - for verification 53
- problems with installation 27
- product components
 - broker 8
 - checking existence 46
 - online documentation 11
 - runtime 8
 - secondary 9
 - User Name Server 9
- publications
 - DB2 98
 - MQSeries 97
 - MQSeries Everyplace 95
 - MQSeries Integrator 95
 - MQSeries Publish/Subscribe 98
- R**
- requirements
 - hardware 3
 - software 5
- resources, default MQSeries 71
- Results Service 56
 - how it works 57
 - running 56
- running applications 56
- runtime support, installing 1
- S**
- Samples
 - samples 10
 - SDK 10
- schema 73
- Scribble 58
 - how it works 59
 - running 58
- SDK 10
- secondary components
 - samples 9
- secondary components (*continued*)
 - Software Developers' Kit (SDK) 9
- security
 - domains 12
 - principals 12
- service
 - applying maintenance 81
 - DB2 82
 - restoring previous level 82
- silent installation 25
- simple message flow
 - assigning 64
 - creating 63
 - deploying 64
 - MQSeries resources 63
 - testing 65
- softcopy books 96
- software requirements 5
- space requirements 4
- Sun Solaris
 - configuring syslog 44
- SupportPac 99
- syslog configuration
 - Sun Solaris 44
- system setup 3
- U**
- uninstalling MQSeries Integrator 77
- user IDs
 - database authorizations 40
 - setting up 36
- User Name Server 9
- V**
- verification, post-installation 2
- verification applications
 - Postcard 60
 - Results Service 56
 - Scribble 58
- verifying installation
 - applications 56
 - creating MQSeries resources 53
 - preparation 53
 - simple message flow 62
 - verifying the installation 53
 - message set import 54
 - workspace import 54
- W**
- Windows 2000 x
- Windows NT
 - importing and deploying resources 54
 - planning for installation 3

workspace
import 54

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.

GC34-5842-03

