Microsoft® Host Integration Server 2000

MSMQ-MQSeries Bridge Configuration Guide

White Paper

Published: November 2000

Table of Contents

MSMQ-MQSeries Bridge: Configuration Guide

White Paper

Published: November 2000

Introduction

This white paper explains how Microsoft Message Queue Server (MSMQ) integrates with IBM MQSeries using the Microsoft MSMQ-MQSeries Bridge. The MSMQ-MQSeries Bridge enables customers to enjoy the best of both worlds, using the integrated and high-performing facilities of MSMQ on Microsoft Windows NT®, while preserving investments in MQSeries on the non-Windows platforms. Shipping with Microsoft SNA Server 4.0 Service Pack 2 and later versions, the MSMQ-MQSeries Bridge provides a simple, cost-effective solution to application integration that is scalable, reliable and offers high performance.

Originally a Level 8[1] product, Microsoft now owns the rights and continues to support, maintain and enhance the Bridge with a new release planned in the next version of SNA Server (named Microsoft Host Integration Server 2000).

This is a technical document that assumes a working knowledge of MSMQ 1.0. (For more information on MSMQ, see the References section at the end of this document.) However, the document provides explanations for many of the MQSeries terms, so little prior knowledge of this environment is required.

This document explains how to integrate MSMQ with MQSeries. It covers the following several key areas.

- Understanding how the MSMQ-MQSeries Bridge works
- Deployment scenarios
- Configuring and testing the Bridge
- Extending the functionality of the Bridge

[1] Level 8 continues to offer a range of cross-platform MSMQ clients that provide an alternative approach to integration.

Definitions

The MSMQ-MQSeries Bridge documentation uses several terms that may be new to readers. Brief descriptions of some of the more important terms are given below and will make more sense after reading the *How the Bridge Works* section. For additional information on terminology, as well as general MSMQ and MSMQ-MQSeries Bridge information, see the References section at the end of this document.

MSMQ Connector
The MSMQ-MQSeries Bridge is an example[2] of an MSMQ Connector Application. An MSMQ Connector Application allows MSMQ-based applications to send and receive messages from computers using other foreign-messaging systems.

MSMQ Connectors use internal connector queues and a connector application to pass messages between the MSMQ and foreign-messaging environment.

MSMQ Connected Network & Foreign-Connected Network
MSMQ 1.0 incorporates the concept of a *Connected Network*—a collection of computers where any two computers can communicate with each other using the same protocol. MSMQ 1.0 expects foreign-messaging systems to be part of their own connected network. One or more foreign-connected networks can be defined for an MQSeries environment. Each foreign-connected network defines a communication path between MSMQ and an MQSeries Queue Manager. Each of these networks consists of a pair of MSMQ Internal Connector Queues for each MSMQ-MQSeries Bridge.

MSMQ Internal Connector Queue
MSMQ routes messages destined for a foreign-messaging system to an appropriate internal connector queue. An MSMQ Connector application then reads those messages from the internal connector queue, performs conversion and delivers the message to the foreign-messaging system.

Therefore, an Internal Connector Queue is a temporary store for MSMQ messages that are then converted and delivered to a foreign-messaging system. MSMQ messages held in an Internal Connector Queue are stored as persistent data, if the message is transactional, or sent with recoverable delivery.

Each MSMQ-MQSeries Bridge has a pair of internal connector queues, one for transactional MSMQ messages and the other for non-transactional MSMQ messages. When an MSMQ application sends messages to a foreign-messaging system within a transaction, MSMQ routes the message to the transactional connector queue.

Connector queues are implicitly created by MSMQ. They are not registered in the MSMQ Information Store (MSMQ 1.0) or the directory (MSMQ 2.0). These queues are transparent to the MSMQ application sending messages to the foreign queue. For applications, sending a message to a foreign queue is identical to sending a message to an MSMQ queue.

[2] Another example is the MSMQ Exchange Connector that provides a way for Microsoft Exchange users to communicate with MSMQ applications.

It is possible to have more than one MSMQ-MQSeries Bridge within a single foreign-connected network. Where this is the case, MSMQ will distribute messages between all pairs of internal connector queues, enabling each MSMQ-MQSeries Bridge to share the load of MSMQ and MQSeries message translation.

MSMQ Foreign Computer
This computer is part of a foreign-messaging system, but exposed to the MSMQ environment. The term ‘computer’ is misleading, because it does not necessarily equate to a physical computer. For example, each MQSeries Queue Manager exposed to MSMQ is an MSMQ Foreign Computer. However, within MQSeries, it is possible to have more than one Queue Manager on any physical computer.

MSMQ Foreign Queue
This queue is part of a foreign-messaging system, but exposed to the MSMQ environment. For example, an MQSeries Queue is a foreign queue where it is exposed to the MSMQ environment. An MQSeries Queue Manager manages MQSeries Queues.

MQSeries Channel
MQSeries incorporates the concept of a one-way communication path between two points in an MQSeries environment. This path is called a channel. A pair of channels must be defined between two points to provide two-way communication. Channels can run between two MQSeries Queue Managers or between an MQSeries Client and an MQSeries Queue Manager.

A Message Queue Interface (MQI) channel runs between an MQSeries Client and an MQSeries Server.

MQSeries Queue Manager
An MQSeries Queue Manager manages a number of queues and message queue operations. An MQSeries Queue Manager is similar to an MSMQ Computer (PEC, PSC, BSC, Routing Server, Independent Client). Note that the terms MQSeries Queue Manager and MQSeries Server are used interchangeably throughout this document.

MQSeries Client
An MQSeries Client is the client Application Programming Interface (API) of MQSeries. It enables applications to access MQSeries Queues managed by an MQSeries Queue Manager. The MSMQ-MQSeries Bridge is an example of an application that uses the MQSeries Client API to access MQSeries Queues. An MQSeries Client is equivalent to a MSMQ dependent client.

Local Queues, Remote Queues and Transmission Queues
A local queue is a queue managed by an MQSeries Queue Manager. An MQSeries Queue Manager views queues managed by other MQSeries Queue Managers as remote queues.

A transmission queue is a special type of local queue used for temporary storage of messages destined for a remote queue on a remote-queue manager. An MQSeries Message Channel Agent (MCA) transmits the messages to their destination queue when the communication link is available.

The MSMQ-MQSeries Bridge requires two transmission queues, one for transactional messages (normal service) and one for non-transactional messages (high service). When an MQSeries application sends messages that are destined for an MSMQ Queue, MQSeries routes the messages to the appropriate transmission queue, depending on the transactional nature of the message send. The Bridge periodically reads these messages from the transmission queues, converts them into MSMQ messages and delivers these messages to the appropriate MSMQ Queue.

MQSeries requires an alias that maps a transmission queue to a particular MSMQ computer. This allows an MQ Queue Manager to decide which transmission queue to use for messages it has received.

Normal and High Service
MQSeries incorporates the concept of normal and high service to distinguish between transactional and non-transactional messages. In the MSMQ-to-MQSeries direction, the Bridge sends transactional messages using normal service and non-transactional messages using high service.

In the MQSeries-to-MSMQ direction, an MQSeries application can send a message using normal or high service by specifying the appropriate remote queue-manager alias, as follows.

<MSMQ-MQSeries Bridge Windows NT computer Name>	for normal service
<MSMQ-MQSeries Bridge Windows NT computer Name>%	for high service

Using normal service, the Bridge supports the *deliver-once* feature of MSMQ and MQSeries. The Bridge offers improved performance using high service, but a message might be delivered more than once in the event of a system failure during transmission.

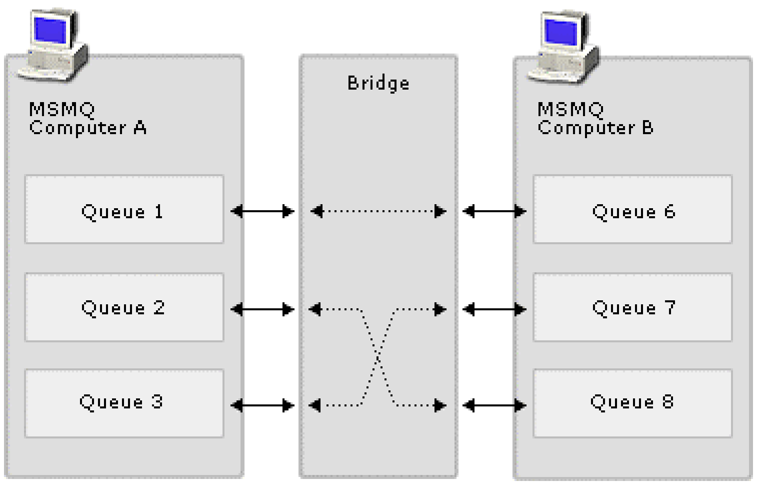
Message Pipes
The MSMQ-MQSeries Bridge features the concept of message pipes. Pipes are communication paths that allow you to transmit messages in various formats. This should not be confused with an MQSeries Channel.

Each Bridge that communicates with an MQSeries Queue Manager has four pipes, two in each direction, with one pipe in each direction for transactional messages and the other pipe in each direction for non-transactional messages.

Pipe Name	Pipe Description
MSMQ -> MQS Normal	Identifies the name of the internal connector queue that the Bridge reads MSMQ transactional messages from to convert and send to MQSeries Queues.
MSMQ -> MQS High	Identifies the name of the internal connector queue that the Bridge reads non-transactional messages from to convert and send to MQSeries Queues.
MQS -> MSMQ Normal	Identifies the name of the transmission queue that the Bridge reads MQSeries messages from to convert and send to MSMQ transactional Queues.
MQS -> MSMQ High	Identifies the name of the transmission queue that the Bridge reads MQSeries messages from to convert and send to MSMQ non-transactional Queues.

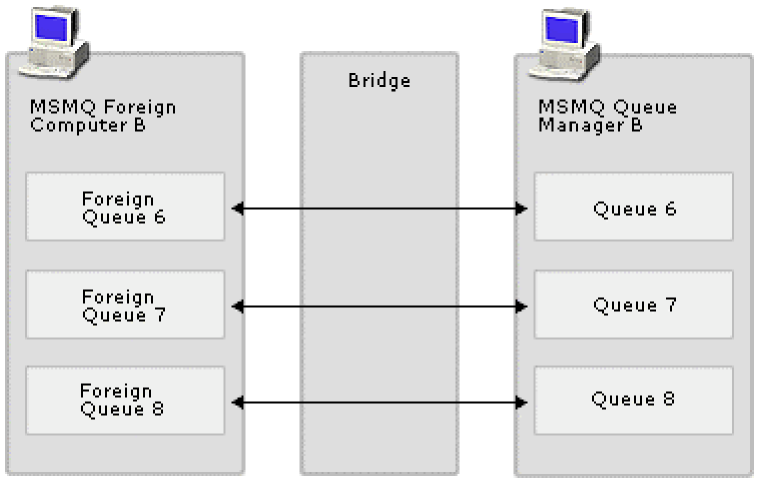
How the Bridge Works

In explaining the properties and attributes of a device, it sometimes helps to know what the device is NOT. The MSMQ-MQSeries Bridge is NOT a switch box. You do not create both MSMQ Queues and MQSeries Queues and then wire the two together.



The Bridge Is NOT a Switchbox

The MSMQ-MQSeries Bridge IS an interface between MSMQ and MQSeries that allows MSMQ applications send messages to MQSeries queues, and allows MQSeries applications send messages to MSMQ queues.



MQSeries Queues Exposed to MSMQ

The philosophy behind the Bridge is to expose queues in each messaging system to those in the other, in terms that are native to each messaging system. The Bridge translates and maps the fields and values of the sending environment to the fields and values of the receiving environment. After mapping and conversion, the MSMQ-MQSeries Bridge then routes the message between the two messaging systems.

- MQSeries Queue Managers and Queues are exposed to the MSMQ environment by defining MQSeries Queue Managers and Queues within MSMQ Explorer as foreign computers and foreign queues, respectively.
- MSMQ Computers and Queues are exposed to the MQSeries environment by creating and importing MQSeries definition files into the configuration of an MQSeries Queue Manager. The definitions identify the transmission queue into which MQSeries should place messages that are destined for an MSMQ Queue.
- The MSMQ-MQSeries Bridge operates transparently allowing MSMQ and MQSeries applications to use their native APIs to deliver messages between the two environments. Neither application is aware that it has crossed between the two environments.

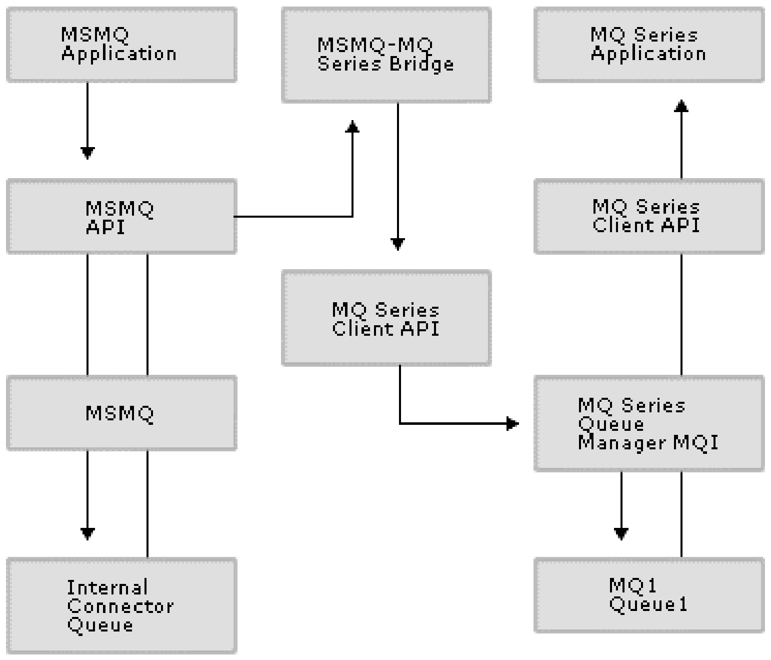
Sending Messages from MSMQ to MQSeries Queues

To send a message from MSMQ to MQSeries, the name of the MQSeries Queue Manager and the name of the MQSeries Queue must be defined within the MSMQ environment using the MSMQ Explorer. This allows MSMQ applications to address MQSeries Queues using MSMQ Queue addressing conventions.

When an MSMQ application sends a message to a queue that is actually an MQSeries Queue, MSMQ transparently routes the message to the internal connector queue used by the MSMQ-MQSeries Bridge. The Bridge reads the message from the internal connector queue, converts the properties of the MSMQ message to equivalent MQSeries message properties, and then sends the MQSeries message to the MQSeries Queue Manager using the MQSeries Client API. MQSeries then handles the transmission of that message to its final destination.

In the MSMQ-to-MQSeries direction, the Bridge thus behaves as an MQSeries Client and uses the MQSeries Client API to submit messages to MQSeries.

The diagram below shows the steps executed when sending an MSMQ message to an MQSeries Queue called Queue1 on an MQSeries Queue Manager called MQ1.



1) MSMQ Application sends a message to "MQ1\Queue 1".

2) MSMQ routes message to the Bridge internal connector queues, because "MQ1\Queue 1" is a foreign queue on a foreign computer.

3) Bridge reads message and converts it to an MQSeries message.

4) Bridge sends message to MQ1 using "MQPut".

5) MQI Channel used for communication between Bridge and MQ1.

6) MQSeries Application reads message from "MQ1 Queue1" using "MQGet".

7) MQI Channel used for communication between MQSeries Application and sending messages from MQSeries to MSMQ Queues.

To send a message from MQSeries to MSMQ, the following MQSeries definitions are required.

- MQSeries Transmission Queues—from which the Bridge reads messages that are destined for an MSMQ Queue.
- Queue Manager and Queue Aliases—for each MSMQ Computer and MSMQ Queue exposed to the MQSeries environment. These allow the MQSeries applications to address MSMQ Queues using MQSeries addressing conventions. MQSeries also uses the aliases to identify the transmission queue into which it must place messages that are destined for the MSMQ environment.
- Channel definitions—between the Bridge and the MQSeries Queue Manager. These channel definitions manage the transmission queues used by the Bridge.

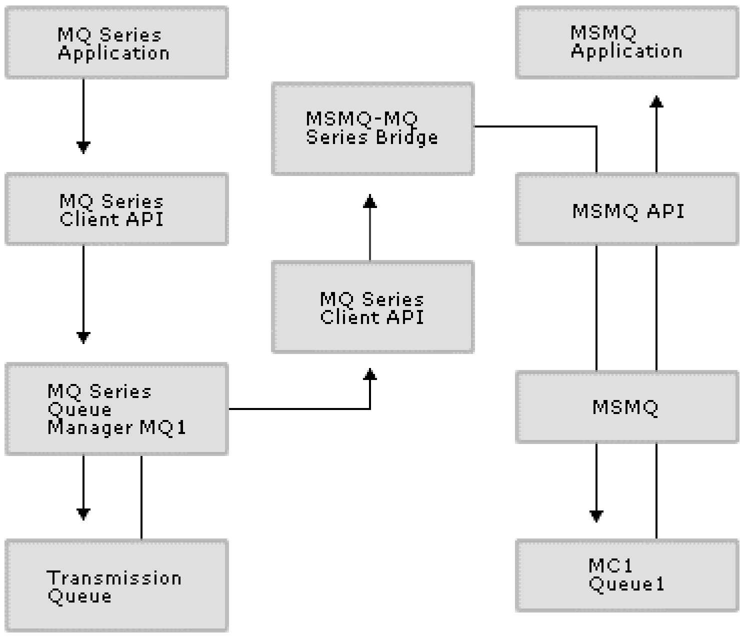
When an MQSeries application sends a message (using MQPUT[3]) to a queue that is actually an MSMQ Queue, MQSeries routes the message to the appropriate transmission queues used by the MSMQ-MQSeries Bridge. The Bridge periodically reads messages from the MQSeries transmission queues using the MQSeries Client API, converts them into MSMQ messages, and then sends the messages on to the appropriate MSMQ Queue. MSMQ then handles the transmission of that message to its final destination.

In the MQSeries-to-MSMQ direction, the Bridge thus behaves as an MQSeries Queue Manager. It uses the MQSeries Client API to read messages from transmission queues, and submit messages to MSMQ.

The diagram below shows the steps executed when sending an MQSeries message to an MSMQ Queue called Queue1 on an MSMQ Computer called MC1 (on an MSMQ-MQSeries Bridge Installation).

The frequency with which the MSMQ-MQSeries Bridge checks transmission queues and internal connector queues is configurable. This is described later.

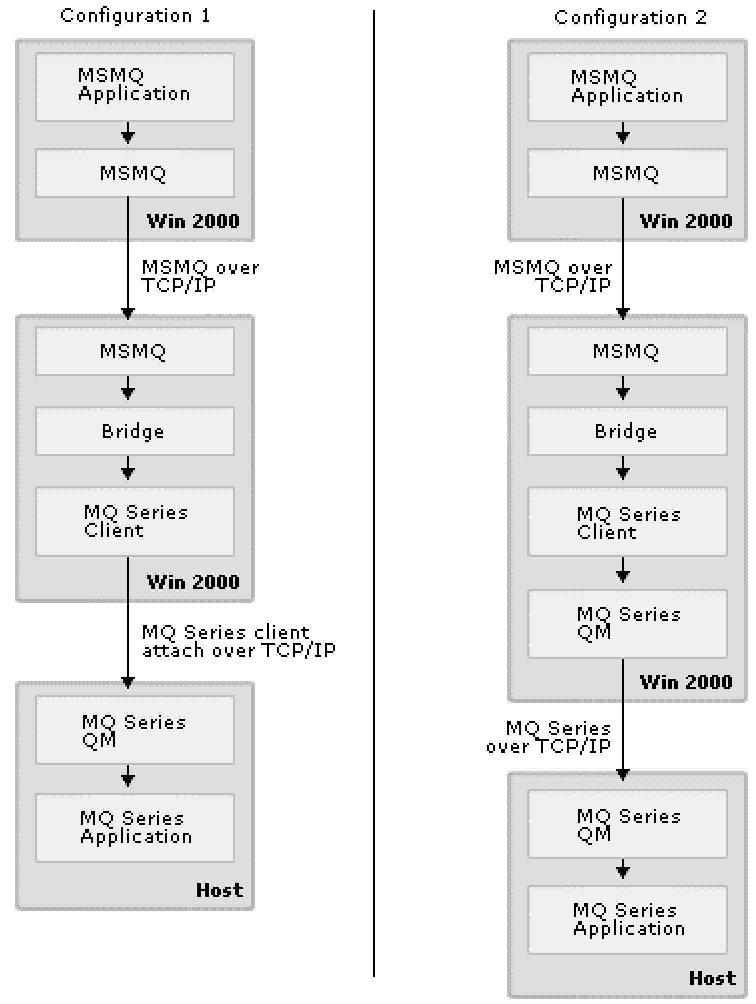
[3] MQGET and MQPUT are part of the MQSeries Client API. MQGET is used for receiving messages and MQPUT is used for sending messages.



1) MQSeries Application sends a message to "MC1 Queue 1" using "MQPut".
2) MQI Channel used for communication between MQSeries Application and MQ1.
3) MQSeries routes message to a transmission queue, because the message is destined for a remote-queue manager (the Bridge).
4) Bridge reads messages from transmission queue held on MQ1.
5) MQI Channel used for communication between Bridge and MQ1.
6) Bridge converts message to an MSMQ message and sends the message to "MC1\Queue1".
7) MSMQ Application reads message from "MC1\Queue1".

MSMQ-MQSeries Bridge Installation

Several possible configurations are available with the Bridge, two of which are shown below.



Several Possible Bridge Configurations

Configuration 1

Connectivity between MSMQ and MQSeries is achieved using the MSMQ-MQSeries Bridge and the MQSeries Client API. The MQSeries Client communicates with the host MQSeries Queue Manager using the Client Attachment feature and a host TCP/IP stack. The Client Attachment feature enables MQSeries Clients to connect directly to the host.

This configuration is the simplest to set up, provided that the Client Attachment feature and a TCP/IP stack are available in the host environment. This configuration is often suitable for many deployments. At the time of writing, the Client Attachment feature was only available in the MVS environment.

Configuration 2

In this configuration, the MQSeries Client used by the MSMQ-MQSeries Bridge connects to a local MQSeries Queue Manager, which then communicates with the host MQSeries Queue Manager through a TCP/IP stack.

Using a local MQSeries Queue Manager on the same server as the Bridge offers improved resilience, improved performance and a configuration that is easier to validate.

- The combination of the Bridge and a Queue Manager installed on the same server improves resilience and effectively achieves the independent-client functionality of MSMQ for MQSeries. An independent client provides the advantage of a 'store-and-forward' capability, which allows a delayed delivery of messages to the host. This means that a temporary loss of the communication link with the host will not result in lost messages.
- The combination of the Bridge and a Queue Manager installed on the same server provides Queue-Manager-to-Queue-Manager communication. This results in greater performance than with the Client Attachment feature.
- By creating MQSeries Queues on the local Queue Manager, it is possible to validate the configuration of the Bridge without requiring assistance from host-support teams.

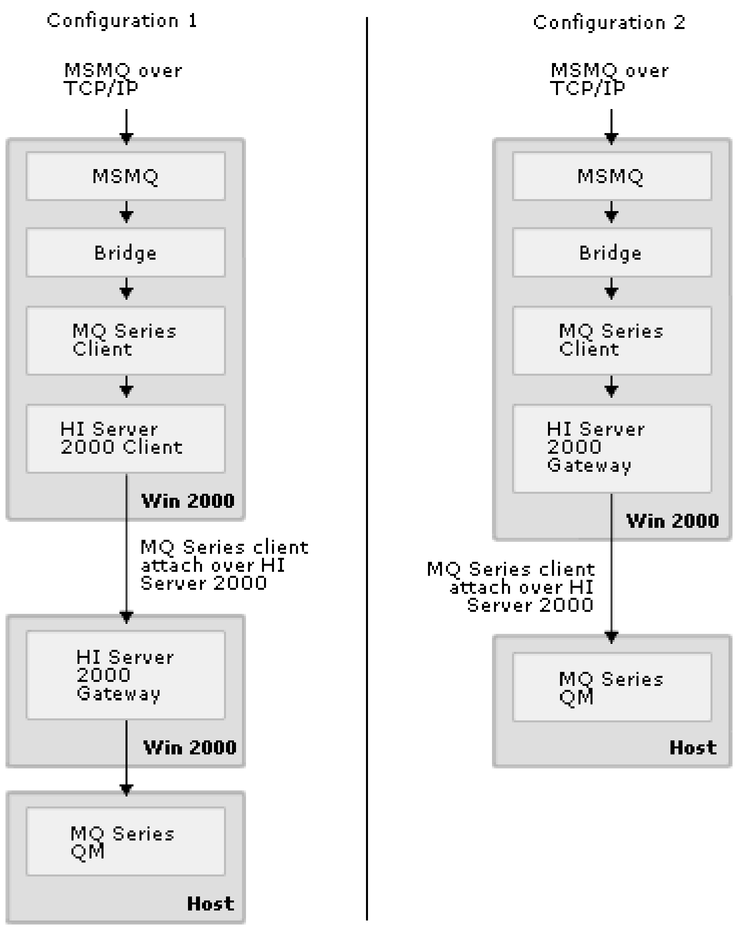
This also provides an alternative approach if the MQSeries Client Attachment feature is not available in the host environment.

SNA Configurations

You can also use the Systems Network Architecture (SNA) protocol between Windows NT and the host. This provides an alternative where TCP/IP access is not available to the host.

These diagrams show possible configurations using an SNA Client and an SNA Server[4].

[4] Microsoft SNA Server or any other SNA server product supported by MQSeries.



Possible Configurations Using an SNA Client and an SNA Server

The decision to use an SNA Client or SNA Server on the same Windows NT server as the Bridge will depend on what SNA infrastructure is already in place.

- Using an SNA Server on the same Windows NT server as the Bridge is likely to offer improved performance where the environment is network bound.
- Using an SNA Client on the same Windows NT server as the Bridge enables load balancing of SNA sessions. It is also easier to establish within existing SNA infrastructure and applications.

Note that SNA Server and the MSMQ-MQSeries Bridge must be installed on Windows NT servers that are part of the same Windows NT Domain.

As discussed previously, you can also use a local MQ Series Queue Manager (as shown in Configuration 2) in conjunction with an SNA Client and SNA Server (as shown above).

MSMQ Considerations

The MSMQ-MQSeries Bridge requires an MSMQ Server. The MSMQ Server can be a Primary Enterprise Controller (PEC), Primary Site Controller (PSC), Backup Site

Controller (BSC) or Routing Server. The actual MSMQ Server used with the Bridge will depend on the MSMQ topology in your environment.

Using an MSMQ Routing Server for the Bridge is the easiest to configure because a Routing Server does not have its own MQIS. This means it does not need SQL Server. The Routing Server will need resilient connectivity to a site controller to resolve MQIS lookups.

It is also easier to configure a fault-tolerant, load-balanced environment where an MSMQ Routing Server is used for the Bridge. Configuring a resilient Bridge environment is discussed later in this document.

Recommended Configuration

From the many configuration choices offered, Configuration 2 together with an MSMQ Routing Server couples good performance with a simple, cost-effective solution. This configuration makes it easy to provide additional capacity and resilience.

MSMQ-MQSeries Bridge configurations that provide fault-tolerance are described later in this document.

Software Installation Order

This is the recommended software installation order for a non-clustered MSMQ-MQSeries Bridge when using the TCP/IP protocol.

Windows NT 4.0 Installation Order

- Microsoft Windows NT Enterprise Edition, version 4.0.
- Microsoft Windows NT Enterprise Edition Service Pack 3.
- Microsoft SQL Server 6.5 or 7.0 (optional).
- Microsoft MSMQ Server, version 1.0.
- Microsoft SQL Server 7.0 (optional).
- IBM MQSeries for Windows NT.
- Microsoft Windows NT Enterprise Edition Service Pack 4.
- Microsoft MSMQ–MQSeries Bridge (part of SNA Server 4.0 Service Pack 2 and later versions).

Windows NT Enterprise Edition 4.0 must be used with the MSMQ-MQSeries Bridge. This is because MSMQ Connector functionality is only available on Windows NT Enterprise Edition. This restriction will be lifted in Microsoft Windows® 2000. Any Windows 2000 Server (Server, Advanced Server, Data Center) is capable of running the MSMQ-MQSeries Bridge.

MSMQ 1.0 will install a limited version of SQL Server 6.5 for the MQIS, as part of its installation. If a full version of SQL Server 6.5 is needed, you must install SQL Server 6.5 before installing MSMQ.

When SQL Server 7.0 is used for the MQIS, SQL Server 7.0 can be installed before installing MSMQ. While SQL Server 7.0 is preferred for many reasons, such as performance and scalability, it does have a few minor issues with MSMQ. One issue can be avoided by simply not installing SQL 6.5 on the system before installing SQL 7.0 and MSMQ. This is outlined in the article "Q214710 BUG: MSMQ PEC, PSC, BSC Setup Fails with SQL Server 7.0". If both SQL 6.5 and 7.0 are used on the system, install MSMQ to the 6.5 installation of SQL then use the SQL Upgrade Wizard to migrate the MQIS to SQL Server 7.0. The other issue with SQL 7.0 and MSMQ is the fact that MSMQ 1.0 Servers and SQL Server 7.0 cannot be used on the same cluster.

As discussed previously, the MSMQ-MQSeries Bridge is capable of working with a variety of different MSMQ Server installations.

Where the SNA protocol is used between the Bridge computer and an MQSeries Queue Manager, an SNA Client or Server must be installed before installing the MSMQ-MQSeries Bridge. Where Microsoft SNA Server is used, this must be the same version from which the MSMQ-MQSeries Bridge is installed.

As discussed later in this document, it is possible to override the conversions that the MSMQ-MQSeries Bridge performs when converting messages from MSMQ to MQSeries. If conversion override is needed with MQSeries version 5.0 or 5.1, install a separate hot-fix (QFE 4122 resolves issues with this functionality). QFE 4122 is also part of SNA Server 4.0 Service Pack 3.

Windows 2000 Installation Order

- Windows 2000 Server (in an Active Directory environment).
- MSMQ (if not installed as part of the Windows 2000 installation).
- IBM MQSeries Version 5.1.
- Microsoft MSMQ–MQSeries Bridge (part of SNA Server 4.0 Service Pack 3).

As previously mentioned, any Windows 2000 Server is capable of running the MSMQ-MQSeries Bridge. SQL Server is not required for MSMQ on Windows 2000 since MQIS functionality is provided by integration with Active Directory.

Licensing

The software licenses required for the MSMQ-MQSeries Bridge are as follows.

- One Microsoft Windows NT 4.0 Enterprise Edition License or Windows 2000 Server License;
- One Microsoft SNA Server 4.0 License;
- One IBM MQSeries MVS License;
- One IBM MQSeries Client Attachment Feature MVS License; OR,
- One IBM MQSeries Server NT License;
- One Microsoft SQL Server 6.5/7.0 License (optional and dependent on the MSMQ Server used).

Whether the SNA or TCP/IP protocol is used between MQSeries on Windows NT and MQSeries on the host, an SNA Server license is always required for the MSMQ-MQSeries Bridge.

Where the MSMQ-MQSeries Bridge uses a local MQSeries Client to communicate directly with the host, a license is required for the MQSeries Client Attachment Feature. For the purposes of this license, the MSMQ-MQSeries Bridge counts as a single user.

Where the MSMQ-MQSeries Bridge uses a local MQSeries Server (Queue Manager) to communicate with the host, a license is required for MQSeries Server NT.

A SQL Server license may also be required, depending on the type of MSMQ Server installed for the MSMQ-MQSeries Bridge. Although MSMQ ships with a limited SQL Server 6.5 license, using a full version of SQL Server 6.5 can improve MSMQ performance. A SQL Server 7.0 license is always required where SQL Server 7.0 is used for the MQIS.

No SQL Server license is required for MSMQ on Windows 2000.

Configuring MSMQ-MQSeries Bridge on Windows NT 4.0

After installing the software required for the MSMQ-MQSeries Bridge, the MSMQ-MQSeries Bridge is ready for configuration.

Configuring the MSMQ-MQSeries Bridge is a three-stage process. It requires configuring MSMQ 1.0, the MSMQ-MQSeries Bridge and MQSeries, in that order.

Configuring MSMQ 1.0

This section describes the steps to configure MSMQ for use with the MSMQ-MQSeries Bridge. The MSMQ Explorer is used for all MSMQ configuration tasks. Familiarity with the MSMQ Explorer is assumed.

Once MSMQ 1.0 has been configured, MSMQ applications can address MQSeries Queues, and MSMQ can route messages intended for MQSeries to the correct MSMQ Internal Connector Queue. However, MSMQ messages will not be converted and delivered to MQSeries. This is because the MSMQ-MQSeries Bridge will not know the MSMQ Internal Connector Queues from which it should read MSMQ messages.

Define a Foreign-Connected Network

MSMQ 1.0 incorporates the concept of a 'Connected Network'. This is defined as: "A collection of computers where any two computers can communicate with each other using the same protocol."

MSMQ 1.0 expects foreign-messaging systems to be part of their own connected network. Therefore, a foreign-connected network[5] must be defined for MQSeries

[5] In Windows 2000, a foreign-connected network is known as a foreign site.

using the MSMQ Explorer. Once a foreign-connected network is defined, MSMQ Internal Connector Queues are automatically created when the MSMQ NT Service is restarted. These internal connector queues are used by an MSMQ-MQSeries Bridge. They act as the route into an MQSeries foreign-connected network.

A foreign-connected network enables the following.

a) A definition of MQSeries Queue Managers and Queues in the MSMQ environment.

b) Notification to MSMQ that sending MSMQ messages to MQSeries requires routing messages to an Internal Connector Queue for processing by an MSMQ Connector application (the Bridge).

c) A definition of an MSMQ Connector (the Bridge).

It is possible to have more than one foreign-connected network. It is also possible to use the same MSMQ-MQSeries Bridge on more than one foreign-connected network. It is important to ensure that the MSMQ-MQSeries Bridge is communicating with different MQSeries Queue Managers through different MQI Channels.

To define a Foreign-Connected Network

1. Open MSMQ Explorer.
2. In the MSMQ Explorer, right-click the **Enterprise** icon. Point to **New**, and click **CN**.
3. Enter the **Connected Network Name**
4. For **Protocol**, select **Foreign**, and then click **OK**.
5. To set the default security privileges of the Connected Network, expand the **Connected Network** folder.
6. Right-click the foreign CN, click **Properties**, click the **Security** tab, and then click **Permissions**.
7. For the **Everyone** group, set the **Type of Access** to **Special**.
8. Select the **Open Connector** check box and click **OK**.

Although the MSMQ-MQSeries Bridge Online Guide implies that the foreign-connected network and the MQI Channel must be the same name, this is not required. The foreign-connected network name can be any name that is valid for MSMQ and MQSeries; the name of the foreign-connected network defaults to the first part of the MQSeries-transmission-queue names used by the MSMQ-MQSeries Bridge.

Add the MSMQ Server to the Foreign-Connected Network

The MSMQ Server[6] that is installed on the same Windows NT server as the MSMQ-MQSeries Bridge must be added to the foreign-connected network using the MSMQ Explorer.

[6] Primary Enterprise Controller, Primary Site Controller, Backup Site Controller or Routing Server.

To add the MSMQ Server to the Foreign-Connected Network

1. Open the MSMQ Explorer.
2. Expand the **Sites** folder, and then click on the name of the site to display the Windows NT computer where the MSMQ-MQSeries Bridge is installed.
3. Right-click the site icon. Point to **Properties**, click on the **Network** tab, and click **Add**.
4. In the **Connected Network Name** dialog box, select the foreign CN name and click **OK**.
5. Repeat this step for each foreign-connected network in which the MSMQ Server is to be a part.
6. In the Windows NT Control Panel, double-click the **Services** icon and then stop and start the Microsoft Message Queue Server.
7. Where multiple MSMQ-MQSeries Bridges are being installed, repeat this process for each MSMQ Server that will be used by each Bridge.

Once the MSMQ Server has been added to the foreign-connected network, the MSMQ NT Service must be cycled. Upon restart of the MSMQ NT Service, the MSMQ Server automatically configures itself as an MSMQ Connector computer. A pair of connector queues is created for each MSMQ Server, one queue for transactional and one queue for non-transactional messages.

The MSMQ Server is now part of the MSMQ connected network and the foreign-connected network for MQSeries.

If the MSMQ Server could not be added to the foreign-connected network, then it is highly likely that the MSMQ Server has not been installed on Windows NT Enterprise Edition, as is required. MSMQ Servers on Windows NT Enterprise Edition are the only version that enables MSMQ Servers to be added to foreign-connected networks. If this is the case, the only solution is to re-install, starting with Windows NT Enterprise Edition.

Define the MQSeries Queue Managers

Each MQSeries Queue Manager handles MQSeries Queues that communicate with an MSMQ application. These MQSeries Queues must be defined within the MSMQ environment using the MSMQ Explorer.

Each MQSeries Queue Manager is defined as an MSMQ foreign computer that is part of the previously defined foreign-connected network. The name of the MSMQ foreign computer must be the same name as the actual MQSeries Queue Manager.

To define an MSMQ Foreign Computer

1. Open the MSMQ Explorer.
2. Expand the **Sites** folder to display the Windows NT computer where MSMQ-MQSeries Bridge is installed.

3. Right-click the **Sites** icon. Point to **New**, and then click **Foreign Computer**.
4. Enter the foreign computer name. The name must be the same as that of the MQSeries Queue Manager.
5. Select the foreign CN where the foreign computer will be placed.
6. Click **Add**, and then click **OK**.

Many MQSeries Queue Managers can be defined within the same foreign-connected network; however, the MSMQ-MQSeries Bridge only communicates directly with one of the MQSeries Queue Managers on each foreign-connected network. The Bridge addresses the other MQSeries Queue Managers through the directly connected MQSeries Queue Manager.

Define the MQSeries Queues

Each MQSeries Queue with which an MSMQ application wants to communicate must be defined within the MSMQ environment using the MSMQ Explorer. Each MQSeries Queue is added as an MSMQ foreign queue to the MSMQ foreign computer (MQSeries Queue Manager) which manages that queue.

To define an MSMQ Foreign Queue

1. Open the MSMQ Explorer.
2. Expand the **Sites** folder to display the MSMQ foreign computer.
3. Right-click the MSMQ foreign computer icon. Point to **New**, and then click **Queue**.
4. Enter the foreign queue name. The name must be the same as that of the MQSeries Queue.
5. Click **Add**, and then click **OK**.

Note that adding an MSMQ foreign queue does not result in creating the MQSeries queue. It simply defines a corresponding name in the MSMQ environment that enables MSMQ applications to address messages to that queue. The actual MQSeries queue must still be created within MQSeries.

Once defining the MSMQ foreign computer and MSMQ foreign queue, MSMQ applications can address messages to the MQSeries environment using the convention "Foreign Computer\Foreign Queue."

Configuring MSMQ-MQSeries Bridge

This section describes the steps to configure the MSMQ-MQSeries Bridge. The MSMQ-MQSeries Bridge Explorer is used for these configuration tasks. During configuration, names will be specified for MQSeries entities, such as transmission queues and channels. Ensure that all names defined for MQSeries are valid within the MQSeries environment and conform to the MQSeries naming conventions used within your organization. Note that MQSeries expects all names to be upper case.

Once the MSMQ-MQSeries Bridge has been configured, MSMQ messages will be read from the MSMQ Internal Connector Queues and converted to MQSeries messages. The MSMQ-MQSeries Bridge will attempt to supply the converted message to MQSeries, but this will fail because MQSeries queue and channel definitions have not been configured.

Define a MQI Channel

The MSMQ-MQSeries Bridge uses the MQSeries Client to communicate with MQSeries. To communicate with MQSeries, a channel must be defined. The channel sets the communication path between the MQSeries Client and the MQSeries Queue Manager. A channel definition is always required regardless of whether the MQSeries Queue Manager is on the same Windows NT server as the MSMQ-MQSeries Bridge.

To define a MQI Channel in the MSMQ-MQSeries Bridge

1. Open the MSMQ-MQSeries Bridge Explorer.
2. Right-click the MSMQ-MQSeries Bridge Explorer icon and select **Properties**.
3. Select the **MQI Channels** tab, and then select **Add**.
4. On the **General** tab of the **Channel Properties** window, specify the following options.

   Channel Name
   Specify a legal MQSeries name for the MQI channel. This is the name that will appear in the channel-definition files that are exported using the MSMQ-MQSeries Bridge Explorer.

   MQSeries Queue Manager
   Specify the name of the MQSeries Queue Manager to which the MQSeries Client connects.

   Transport Type
   TCP/IP or SNA. Specify the protocol used to communicate between Windows NT and the host.
5. On the **Address** tab of **Channel Properties**, specify the TCP/IP Address and Port number of the MQSeries listener (default port used by MQSeries is 1414). This specifies the end-point at which the MQSeries Queue Manager is listening. The MQSeries Client will use this address to establish communication with the MQSeries Queue Manager.
6. For SNA, specify the **Side Information Record**. This is the CPI-C Symbolic Name defined in SNA Server. It specifies the SNA end-point at which the MQSeries Queue Manager is listening.
7. On the **Security** tab of **Channel Properties** specify the **MCA User**. Use an existing or new MQSeries user name. This is the name under which the server side of the MQI channel runs.
8. Repeat these steps for each MSMQ-MQSeries Bridge. Remember that the MSMQ-MQSeries Bridge has point-to-point communication (using an

MQI Channel) with a single MQSeries Queue Manage for each foreign-connected network.

If an MCA User is not specified, the server side of the channel runs under the default user name. This name is the value of the MQSeries `SYSTEM.DEF.SVRCONN` parameter. If an MCA User is specified, then the MQSeries Queues that this MCA User can access must be specified in MQSeries.

Add a Foreign-Connected Network to the MSMQ-MQSeries Bridge

The MSMQ Server was added to the foreign-connected network as part of the MSMQ configuration. We have also defined the MQI Channel that the MSMQ-MQSeries Bridge should use to communicate with MQSeries. However, the association has not been established between the MSMQ internal connector queues, the MSMQ-MQSeries Bridge and the particular MQI channel used to send messages to MQSeries. This configuration step creates this association.

To Add a Foreign-Connected Network

1. Within the MSMQ-MQSeries Bridge Explorer, right-click the MSMQ-MQSeries Bridge to which the connected network (CN) is connected. Click **New CN**, and select the CN name from the list.
2. Repeat for each foreign-connected network of which the MSMQ-MQSeries Bridge is a part.

Adding a foreign-connected network to the MSMQ-MQSeries Bridge creates four message pipes. For details on message pipes, refer to the **Definitions** section of this document.

Once a foreign-connected network has been added to the MSMQ-MQSeries Bridge, additional configuration information is needed. Right-click the CN and select Properties. The following information must be specified.

MQSeries Queue Manager Name

This is the name of the MQSeries Queue Manager that was specified when defining the MQI Channel in the previous step. Where the MSMQ-MQSeries Bridge is part of more than one foreign-connected network, ensure that different MQSeries Queue Managers are used for each CN.

Reply To Queue Manager Name

This is the MQSeries Queue Manager name to which MQSeries will return report and acknowledgement messages. By default, this is the Windows NT computer name on which the MSMQ-MQSeries Bridge is installed.

Remember that the MSMQ-MQSeries Bridge appears to MQSeries as an MQSeries Queue Manager, and is exposed to MQSeries by means of Queue-manager aliases. The name specified here is used by MQSeries to determine the transmission queue into which to place report and acknowledgement messages. Specifying the MSMQ-MQSeries Bridge Windows NT computer name ensures that these messages get back to the MSMQ-MQSeries Bridge for processing.

Remember also that different queue-manager aliases are used for normal and high service. By using the Windows NT computer name, report and acknowledgement messages will be sent using normal service. By adding a percent sign to the Windows NT computer name entered into this field, report and acknowledgement messages are sent using high service.

It is also possible to redirect report and acknowledgments by specifying a different alias. For example, to receive acknowledgments, through a different MSMQ-MQSeries Bridge connected to the same MQSeries Queue Manager, enter the MQSeries alias for the different Bridge computer.

Message Pipes

Each message pipe is individually configurable. Right-click a message pipe and select **Properties** to change the settings.

Batch Settings

To optimize performance, the MSMQ-MQSeries Bridge batches messages together. A number of batch settings control the size of a batch. Once a batch size has been reached, the MSMQ-MQSeries Bridge checks that the batch was successfully received.

Increasing the batch size may improve performance, but may also increase the quantity of retransmitted data after a communication failure.

Cache Settings

To optimize performance, the MSMQ-MQSeries Bridge will cache queue handles to reduce the overhead of opening and closing a queue for each message. The Cache Settings determine the length of time the MSMQ-MQSeries Bridge will cache queue handles before closing a queue.

Export the MQSeries Definition Files

The majority of the configuration within the MSMQ-MQSeries Bridge Explorer is used to create MQSeries definition files that can be exported from the MSMQ-MQSeries Bridge Explorer and then used to configure MQSeries.

Exporting the MQSeries definition files from the MSMQ-MQSeries Bridge Explorer is the easiest and most reliable way of defining the MQSeries entities required by the MSMQ-MQSeries Bridge.

To Export the MQSeries Definitions

1. Open the MSMQ-MQSeries Bridge Explorer.
2. Right-click the MSMQ-MQSeries Bridge and click Export Client Definitions.
3. In the dialog box, specify a directory to store the Clientdf.txt definition file.
4. Right-click the MSMQ-MQSeries Bridge icon and click **Export Server Definitions**.
5. In the dialog box, specify a directory to store the `<QM name>.txt` definition files.
6. Repeat steps 1-5 for each MSMQ-MQSeries Bridge in your network, storing each set of definition files in a separate directory.

The exported client-definition file configures the client side of the MQI channel that the Bridge uses to communicate with the MQSeries Queue Manager. The name of this channel is the name given when defining the MQI Channel within the MSMQ-MQSeries Bridge Explorer.

The exported server-definition file configures the server side of the MQI channel that the Bridge uses to communicate with the MQSeries Queue Manager. The server-definition file also configures transmission queues and queue-manager aliases that MQSeries uses to communicate with the Bridge. The names of the transmission queues are of the following form.

- <Foreign-connected network Name>.XMITQ for normal service.
- <Foreign-connected network Name>.XMITQ.HIGH for high service.

The names of the queue-manager aliases are based on the value specified in the 'Reply To Queue Manager Name' of the connected network properties. By default these are the following.

- <Windows NT computer name> for normal service.
- <Windows NT computer name>% for high service.

The names of the MQI channel and the transmission queues defined in the exported files can be changed to any permitted MQSeries name. Remember that the names that appear in the exported definition files were generated directly from the configuration settings that were specified in the MSMQ-MQSeries Bridge Explorer. Therefore, if the names of the MQI channel or transmission queues are changed, then the configuration settings within the MSMQ-MQSeries Bridge Explorer must also be changed.

The contents of the exported definition files are discussed later in this document (refer to the section MQSeries Definitions Explained). Importing the MQSeries definitions is also covered later in this document.

It is worth noting that the default server definitions generated from the MSMQ-MQSeries Bridge Explorer are the minimum required to configure MQSeries to send messages to MSMQ Computers and MSMQ Queues. Additional server definitions are required to expose specific MSMQ Computers and MSMQ Queues to the MQSeries environment. For details, refer to the section
Configuring MQ Series to Send Messages to Specific MSMQ Queues

Backup the Configuration

The MSMQ-MQSeries Bridge stores its configuration in the registry. It is a good idea to backup the configuration to assist in disaster recovery.

The following registry keys should be backed up as they define the current configuration of the MSMQ-MQSeries Bridge.

For standalone configurations:

HKLM\Software\Microsoft\MQBridge\Server

For cluster configurations:
HKLM\Cluster\Software\Microsoft\MQBridge\Server

Backing up of the MSMQ and MQSeries configuration is also required. Refer to the MSMQ Administrators Guide and appropriate MQSeries documentation.

Knowledge Base articles Q191535 and Q191536 describe backup and restore of the MQIS. An MQIS Restore Wizard is also available from the following.

ftp://ftp.microsoft.com/bussys/distapps/msmq/public-fixes/1.0/mqis-restore/

Configuring MQSeries

The MSMQ-MQSeries Bridge requires the following MQSeries definitions (see below).

- Transmission Queues for normal and high service, used for sending messages from MQSeries to MSMQ.
- Queue-manager aliases for normal and high service, used for sending messages from MQSeries to MSMQ.
- A model queue, used when sending transactional messages from MSMQ to MQSeries.
- An MQI Channel definition (server and client sides) for both directions.

These definitions must be imported into the MQSeries Queue Manager with which the MSMQ-MQSeries Bridge communicates. Importing the definitions generated from the MSMQ-MQSeries Bridge Explorer is the quickest and most reliable way of configuring MQSeries.

Once MQSeries queue and channel definitions have been configured, the MSMQ-MQSeries Bridge can send and receive messages between MSMQ and MQSeries.

To Import a Server-Definition File

1. Transfer the exported server-definition file from the MSMQ-MQSeries Bridge computer to the MQSeries Queue Manager computer.
2. Run the MQSC command to create the definitions in MQSeries. For example:

   ```
   >RUNMQSC <QM Name> <Server-definition file>.TXT REPORT.OUT
   ```

   The REPORT.OUT file should be reviewed for any errors.
3. Repeat steps 1-2 for each server-definition file exported from the MSMQ-MQSeries Bridge Explorer; that is, for each MQSeries Queue Manager connected to the MSMQ-MQSeries Bridge.

To Import a Client-Definition File

1. Transfer the exported client-definition file from the MSMQ-MQSeries Bridge computer to any MQSeries Queue Manager computer.
2. Run the MQSC command to create the definitions in MQSeries. For example:

   ```
   >RUNMQSC <QM Name> <Client-definition file>.TXT REPORT.OUT
   ```

   The REPORT.OUT file should be reviewed for any errors.

   MQSeries creates a channel-definition table file for the client-side MQI channel, and stores the file on the MQSeries Queue Manager computer. The default name of the channel file is AMQCLCHL.TAB. For the directory location, see the MQSeries documentation.
3. Transfer the channel table file to the MSMQ-MQSeries Bridge computer. Store the file in any convenient directory (for example the MQSeries Client directory or the `\windows\system32` directory).
4. Set the following MQSeries environment variables on the MSMQ MQSeries Bridge computer.

   MQCHLLIB. This is the path to the client channel-definition table as per step 3.
   MQCHLTAB. This is the name of the client channel-definition table as per step 2.
   Do not specify a value for the MQSERVER environment variable, as this will override the values of these two environment variables. Combine the channel files where other MQSeries applications exist on the same computer as the Bridge (see the IBM MQSeries documentation).
5. For the environment variables to take effect, reboot the MSMQ-MQSeries Bridge computer.
6. Repeat steps 1-5 for each MSMQ-MQSeries Bridge.

Where the MSMQ-MQSeries Bridge is used in a cluster, follow steps 1-6 above as well as the following steps.

- Store the channel table file on the shared cluster disk, and set the environment variables on each node to point to the shared location.
- Store a separate copy of the channel file on each cluster node, and set the environment variables of each node to its own copy.

The MQSeries Client uses the MQCHLLIB and MQCHLTAB to find and read the channel-definition table. The channel-definition table defines what channel (transport, address and port) should be used to connect[7] to a particular MQSeries Queue Manager.

When the MSMQ-MQSeries Bridge converts an MSMQ message to an MQSeries message, the Bridge uses MQConn() of the MQSeries Client API to connect to the MQSeries Queue Manager. At this point, the client channel-definition table is searched to identify which channel to use for communicating with the MQSeries Queue Manager. If the channel-definition table is not updated using the exported

[7] Remember that it is the channel that is actually used to achieve communication between an MQSeries Client and an MQSeries Server (Queue Manager).

client-definition file, then the MSMQ-MQSeries Bridge will not be able to communicate with the MQSeries Queue Manager.

Transport Considerations

TCP/IP
Where the TCP/IP protocol is used between the MQSeries Client and MQSeries Queue Manager, check the "Keep Alive" and "Keep Alive Time" settings in the MQSeries Queue Manager initialization file (qm.ini). "Keep Alive" should be set to "YES" and the "Keep Alive Time" should be no more than half the message pipes retry delay. This allows the MQSeries Listener to release the resources of a broken connection before the MSMQ-MQSeries Bridge retries the connection.

The message pipes retry delay can be found by right-clicking a message pipe in the MSMQ-MQSeries Bridge Explorer. Select **Properties,** and then select the **Cache** tab.

SNA
The MSMQ-MQSeries Online Guide contains details of how to configure an MQI Channel to use the SNA LU 6.2 protocol. The guide also provides sample APPC/MVS, VTAM and MQSeries Queue Manager configuration values to set-up an MQSeries Listener in an MVS environment that communicates using SNA LU 6.2.

Configuring Security

Authentication is supported from the MSMQ sending application up to the MSMQ-MQSeries Bridge. Authentication from the MSMQ-MQSeries Bridge to MQSeries, or from MQSeries to MSMQ, is not currently supported. MSMQ encryption features are also not currently supported.

MSMQ Security

MSMQ foreign computers and MSMQ foreign queues can be secured using the MSMQ Explorer in the same way as MSMQ computers and queues. This ensures that only certain users can submit MSMQ messages to the MSMQ-MQSeries Bridge, and ultimately to the MQSeries application.

The MSMQ Internal Connector Queues used by the MSMQ-MQSeries Bridge cannot be secured. However, MSMQ applications cannot access and send MSMQ messages directly to these queues.

MSMQ-MQSeries Bridge

No specific security settings surround the MSMQ-MQSeries Bridge. The Bridge will convert MSMQ messages that arrive in the MSMQ Internal Connector Queue and MQSeries messages that arrive in the MQSeries transmission queues.

Security credentials contained in the MSMQ message properties PROPID_M_SENDERXXX are not converted in the MSMQ-to-MQSeries direction. In the MQSeries-to-MSMQ direction, the UserIdentifier field of an MQ Message Descriptor (see later) is converted to the MSMQ message property PROPID_M_SENDERID, but only if it contains a valid NT SID value.

Where SNA Server is used, a Host Account Cache, Windows NT and Host Account Synchronization service are not required, because the MSMQ-MQSeries Bridge does not currently support host-security integration.

To secure the Bridge in the MSMQ-to-MQSeries direction, secure the MSMQ foreign queues. This prevents unauthorized MSMQ messages from arriving in the MSMQ Internal Connector Queues. In the MQSeries-to-MSMQ direction, follow the guidelines in the MQSeries Administrators guide. This prevents unauthorized MQSeries messages from arriving in the MQSeries transmission queues.

MQSeries Security

The MQI Channel definition that links the MQSeries Client and MQSeries Queue Manager has an MCAUSER parameter. The value of this parameter is the MQSeries user under which the server-side of the channel runs.

If an MCA User is specified, then the MQSeries Queues that this MCA User can access must be specified in MQSeries.

If an MCA User is not specified, the server side of the channel runs under the default user name, which is the value of the MQSeries `SYSTEM.DEF.SVRCONN` parameter.

Testing the Configuration

This section works through a series of steps to ensure that the MSMQ-MQSeries Bridge configuration is functioning as expected. It also offers some troubleshooting tips.

The following assumptions are made for this section.

- BRIDGE1 is the Windows NT computer name on which the MSMQ-MQSeries Bridge is installed.
- An MSMQ Queue called MSMQ1 exists on the Bridge computer.
- The name of the MQSeries Queue Manager is MQQM1.
- An MQSeries Queue called MQ1 on MQQM1 has been specifically created for testing the Bridge configuration.
- The MQI Channel Name is MQQM.CLIENT.CHL.

It is often a good idea to relax security before testing the configuration. This will help solve configuration problems.

Sample Programs

Several samples have been provided by the MSMQ-MQSeries Bridge and by MQSeries. Awareness of the following samples will help diagnose configuration problems.

The MSMQ-MQSeries Bridge provides the following.

MQSRRECV	Reads messages from an MQSeries queue.
MQSRSEND	Sends messages to an MQSeries queue.

MSMQRECV	Reads messages from an MSMQ queue.
MSMQSEND	Sends messages to an MSMQ queue.

These can be found in the 'samples' directory under the directory to which the MSMQ-MQSeries Bridge was installed.

MQSeries ships with many sample programs that can be used to test MQSeries functionality. These samples can be found below the directory where MQSeries is installed. A 'File Find' will also locate these sample programs.
Some of the more useful samples include the following.

Amqsputc	Writes a message to an MQSeries queue—similar to MQSRSEND.
Amqsgetc	Reads a message from an MQSeries queue—similar to MQSRRECV.
Amqsbcg	Browses messages on an MQSeries queue—similar to an MSMQ Peek.

In the event of a failure, these programs output error messages together with reason codes. The reason codes can be looked up in the IBM MQSeries documentation to help diagnose the problem.

MQSeries Command Console

MQSeries also has a command console that can be used to perform various configuration and management activities. From a command prompt type:

```
> RUNMQSC
```

This displays MQSeries command console. The 'HELP' command lists the available commands. Awareness and familiarity of this console will help diagnose configuration problems.

Testing the MQSeries Client

On the Windows NT computer (BRIDGE1), run the MQSRRECV program in a command prompt using the following command line.

```
MQSRRECV MQQM1 MQ1
```

MQSRRECV can be used to test connectivity between the MQSeries Client and MQSeries Queue Manager. MQSRRECV waits indefinitely to receive a message from the MQSeries queue MQ1. If the program successfully runs, it displays the following message.

```
Use <CTRL-C> to stop!
```

If the program does not succeed then this indicates the MQSeries Client is unable to communicate with the MQSeries Queue Manager. Check the following.

1. The MQSeries Environment variables MQCHLLIB and MQCHLTAB are defined and point to the location of the client channel-definition table.
2. If the client channel-definition table does not exist or contains no entries, re-generate the table as described in **Importing the Client-Definition File**.

3. The MQSeries Listener is running and listening on the address specified when the MQI Channel was defined in the MSMQ-MQSeries Explorer.

4. Where the MQSeries Queue Manager is on a different computer than the MQSeries Client. Test basic network connectivity (For example, 'ping' for TCP/IP).

5. Override the MQSeries Environment variables by specifying a value for the environment variable MQSERVER, as follows.

   ```
   SET MQSERVER=<channel name>/<transport type>/<connection
   name>[(<port number>)]
   ```

   An example for TCP/IP is:

   ```
   >SET MQSERVER= MQQM.CLIENT.CHL /TCP/10.10.10.5(1414)
   ```

   An example for SNA is:

   ```
   >SET MQSERVER= MQQM.CLIENT.CHL /LU62/MQSCPIC
   ```

   Where MQSCPIC is the side information record name (CPIC Symbolic Destination Name).
6. Check the NT Event Log for any relevant messages.

Re-run the MQSRRECV program after each check.

Testing MQSeries Client to Queue Manager Communication

Perform the following on the Windows NT computer (BRIDGE1).

1. Run the program MQSRRECV in one command prompt using the following command line:

   ```
   MQSRRECV MQQM1 MQ1
   ```

2. Run the program MQSRSEND in a separate command prompt using the following command line:

   ```
   MQSRSEND MQQM1 MQ1
   ```

3. The MQSRSEND program will place lines of text (separated by Carriage Return) into MQSeries messages and send the message to the MQ1 queue. The text should then appear in the MQSRRECV window.

If this test does not work, check the items listed in **Testing the MQSeries Client**.

Testing MSMQ Connectivity

1. Open the MSMQ Explorer on a non-Bridge computer.

2. Right-click the MSMQ Server used by the MSMQ-MQSeries Bridge and select MQPing. After approximately 30 seconds a small green tick should appear beside the MSMQ Server icon.

3. If this does not happen, confirm the following.
   a) The MSMQ Server used by the MSMQ-MQSeries Bridge is part of both the foreign-connected network and the MSMQ connected network.
   b) The foreign-connected network protocol is 'foreign'.
   c) The MSMQ connected network protocol matches the protocol used in your environment (For example, IP).
   d) Basic network connectivity exists between the two Windows NT computers. (For example, use 'ping' to test TCP/IP connectivity)

If MSMQ connectivity exists, then it should be possible for MSMQ to route messages destined for MQSeries to the appropriate internal connector queues.

Sending MSMQ Messages to MQSeries

Open the MSMQ-MQSeries Explorer and expand the tree-view to see all the message pipes. From a command prompt, run:

```
MSMQSend MQQM1\MQ1
```

MSMQSend sends ten messages to the queue specified on the command line. Using the MSMQ-MQSeries Bridge Explorer, the 'Messages Sent' column of the MSMQ->MQS High message pipe should increase by 10.

The MQSeries command line sample amqsbcg can also be used to browse the contents of the MQ1 queue. From a command prompt, run:

```
amqsbcg MQQM1 MQ1
```

This will display the ten messages sent from MSMQSend.

The MQSRRecv sample can be used to receive the messages sent by MSMQSend. The MQSRRecv sample can be used to receive the messages sent by MSMQSend. From a command prompt, run:

```
MQSRRecv MQQM1 MQ1
```

If the 'Q Depth' column of the MSMQ->MQS High message pipe has increased by 10 and does not decrease, this suggests that MSMQ has been able to route the MSMQ messages to the internal connector queue, but that problems lie elsewhere. Possible causes are the following.

1. The MSMQ-MQSeries Bridge is not running.
   a) Check that the NT Service MSMQ-MQSeries Bridge is running and check the NT event log for any messages.
2. The MSMQ->MQS Message Pipe is not running.
   a) Using the MSMQ-MQSeries Explorer, check to ensure that the message pipes are enabled and running.
3. The MSMQ-MQSeries Bridge is having problems communicating with MQSeries. Refer to **Testing the MQSeries Client**.

If the 'Messages Sent' column of the MSMQ->MQS High message pipe increases by 10 but no messages are received using MQSRRecv, this suggests that the

MSMQ-MQSeries Bridge has processed the messages, but that the problems lie elsewhere. Possible causes are the following.

1. The destination specified for the messages is invalid and the messages have actually been sent to the dead-letter queue on the MQSeries side.
    a) Check that the correct queue name was specified in the MSMQSend command line.
    b) Check that the MQSeries queue and queue manager have been properly defined. Use the amqsputc sample to send an MQSeries message to the MQSeries queue without using the Bridge. Although the MQSeries Queue Name and Manager have been defined in the MSMQ Explorer, the MQSeries Queue may not have been defined in MQSeries.
    c) Check that appropriate channel definitions and transmission queues exist to enable MQSeries to route the message to its destination.
    d) Check the IBM MQSeries documentation for the location of the dead-letter queues and check that the messages are there.
2. The MSMQ-MQSeries Bridge is having problems communicating with MQSeries. Refer to **Testing the MQSeries Client**.
3. A conversion error occurred. Check the NT event log.

If the 'Q Depth' column of the MQS->MSMQ High message pipe has increased by 10 and does not decrease, this suggests that MQSeries has been able to route the MQSeries messages to the MQSeries transmission queue, but that problems lie elsewhere. Possible causes are the following.

1. The MSMQ-MQSeries Bridge is not running.

   Check to see whether the Windows NT Service MSMQ-MQSeries Bridge is running and check the Windows NT event log for any messages.
2. The MQS->MSMQ High Message Pipe is not running.

   Using the MSMQ-MQSeries Explorer, check to ensure that the message pipes are enabled and running.
3. The MSMQ-MQSeries Bridge is having problems communicating with MQSeries. Refer to **Testing the MQSeries Client**.

If both the 'Messages Sent' and 'Q Depth' columns of the MQS->MSMQ High pipe do not change, then this suggests the following.

1. The MSMQ-MQSeries Bridge is having problems communicating with MQSeries. Refer to **Testing the MQSeries Client**.
2. MQSeries was unable to route the MQSeries message to the appropriate transmission queue.
    a) Check that the transmission queues used by the MSMQ-MQSeries Bridge are properly defined.
    b) Check that appropriate channel definitions, transmission queues and queue-manager aliases exist to enable MQSeries to route the message to the correct transmission queue.
    c) Check the MQSeries dead-letter queues.

Managing and Monitoring the Bridge

MSMQ-MQSeries Bridge Explorer

The MSMQ-MQSeries Bridge Explorer can be used to perform management, configuration and monitoring of the Bridge.

Performance Monitor Counters

The MSMQ-MQSeries Bridge offers a number of performance counters that enable remote monitoring of the Bridge.

The MSMQ-MQSeries Bridge performance counters enable monitoring of the following.

- Queue Depth and the number of Messages Sent for each MSMQ Internal Connector queue. The internal connector queues are not visible in the MSMQ Explorer. These performance counters and the MSMQ-MQSeries Bridge Explorer are the only way to monitor these queues.
- Queue Depth and the number of Messages Sent for each MQSeries transmission queue.
- Status of each message pipe.
- Status of the MSMQ-MQSeries Bridge.

Configuring MSMQ-MQSeries Bridge on Windows 2000

Configuring the MSMQ-MQSeries Bridge on Windows 2000 is a three-stage process. It requires configuring MSMQ 2.0, the MSMQ-MQSeries Bridge and MQSeries in that order. Configuring MSMQ 2.0 is the only stage that differs from what has already been described for configuring the MSMQ-MQSeries Bridge on Windows NT 4.0.

MSMQ 2.0 and MSMQ-MQSeries Bridge

MSMQ 2.0 is built into Windows 2000 and offers the following additional functionality.

- Integration with Active Directory, precluding the use of a separate SQL Server to maintain the MQIS.
- Mixed-mode operation, enabling MSMQ 1.0 and MSMQ 2.0 environments to co-exist.
- Performance improvements, particularly in the area of transactions.
- Workgroup Mode, enabling Windows 2000 computers to use MSMQ 2.0 without the need for an Active Directory.

The MSMQ-MQSeries Bridge is capable of running on Windows 2000. The MSMQ-MQSeries Bridge can be installed on any of the Windows 2000 Servers (Server, Advanced Server, Data Center).

To use the MSMQ-MQSeries Bridge on Windows 2000, install the following.

- Windows 2000 Server (Server, Advanced Server or Data Center)
- IBM MQSeries Version 5.0 or 5.1
- MSMQ-MQSeries Bridge (from SNA Server 4.0 Service Pack 3)

Additional installation steps for the MSMQ-MQSeries Bridge are documented in the readme.txt.

The following changes in MSMQ 2.0 are relevant to the MSMQ-MQSeries Bridge.

- Connected Networks have been replaced with the term "Site." This is in keeping with Active Directory terminology. Therefore, a foreign-connected network now becomes a foreign site.
- The MSMQ 2.0 COM API has changed. Many more of the MSMQ message properties are now exposed through the MSMQ Component Object Model (COM) API. In particular, the extension property (PROPID_M_EXTENSION) is now accessible from Visual Basic, making it easier to override MSMQ-MQSeries Bridge conversions.
- Creating foreign sites and foreign computers is achieved using the Active Directory Sites and Services MMC snap-in.

Configuring MSMQ 2.0

This section assumes that MSMQ 2.0 has been installed (with routing enabled) onto a Windows 2000 server on which installation of the MSMQ-MQSeries Bridge is planned. Using the Active Directory Snap-in 'Users and Computers', it should be possible to see the MSMQ object.

```
Domain Controllers
    Computer Name (Your computer name)
        MSMQ
```

If this is not the case, check that the view is set to 'Advanced Features'. If the MSMQ object still does not appear, then MSMQ 2.0 is not installed. Check the Windows 2000 Message Queuing Help to ensure that the MSMQ 2.0 pre-requisites are met before installing MSMQ 2.0.

From Control Panel, select **Add Remove Components** and install the Message Queuing Services. During the installation of MSMQ 2.0, ensure that 'Enable Routing' is selected. This will enable defining a foreign site (foreign-connected network) and define routing links to the foreign site, which are required to set-up an MSMQ-MQSeries Bridge.

Active Directory Snap-Ins are used to configure MSMQ 2.0 for use with the MSMQ-MQSeries Bridge. This requires permissions to make changes to your Active Directory. For example, when creating a routing link, the default is that only users of the Enterprise Administrators group can make these changes. Check to ensure that the appropriate permissions are available to make the required Active Directory changes before commencing MSMQ 2.0 configuration.

Define a Foreign Site

1. From the 'Sites and Services' Snap-in, select **View** and **Show Services Node**.
2. Under **Services**, right click **MsmqServices** and select **New Foreign Site**.
3. Enter the name for the site. This is the name of the foreign-connected network in MSMQ 1.0. The same naming conventions apply to the site name as to the foreign-connected network name for MSMQ 1.0.

Define the MQSeries Queue Managers

1. From the 'Sites and Services' Snap-in, select **View** and **Show Services Node** (if not already selected).
2. Under **Services**, right click **MsmqServices** and select **New** followed by **Foreign Computer**.
3. Enter the name for the foreign computer. The name must be the same as that of the MQSeries Queue Manager.
4. Enter the name for the site. This is the name of the foreign site that was just created.

Define the MQSeries Queues

1. From the 'Users and Computers' Snap-in, select **View** and **Advanced Features** followed by **View and Users**, **Groups and Computers as containers**.
2. Expand the Computers folder and select the foreign computer that was just created.
3. **Right click** the **MSMQ** object and select **New** followed by **MSMQ Queue**.
4. Enter the name for the queue. The name must be the same as that of the MQSeries Queue.

Create a Routing Link

A routing link is required to enable MSMQ 2.0 to route messages between the current Windows 2000 site and the newly created foreign site that is used for MQSeries. Use the following steps to create a routing link.

1. From the 'Sites and Services' Snap-in, select **View** and **Show Services Node** (if not already selected).
2. Under **Services**, right click **MsmqServices** and select **New** followed by **MSMQ Routing Link**.
3. Set Site 1 to the name of the foreign site that was just created. Set Site 2 to the name of the current Windows 2000 site.
4. Set the routing link cost to 1. This should be the default value. Using a value greater than 1 is only relevant where multiple routing links are

defined between sites and one route must be enforced over another. Do not set this value to zero as this effectively means no link. Routing of MSMQ messages to the foreign site will fail if it is set to zero.

Define a Site Gate

A site gate is an MSMQ Server that is configured to route messages between sites on behalf of other clients. The MSMQ Server that will be defined as the site gate is the computer that will run the MSMQ-MQSeries Bridge. This site gate will use the routing link that was just created.

To define a Site Gate

1. From the 'Sites and Services' Snap-in, select **View** and **Show Services Node** (if not already selected).
2. Under **Services**, right click **MsmqServices**. The routing link that was just created should appear in the right pane window.
3. **Right-click** the routing link and select **Properties**.
4. Select the **Site Gates** tab. In **Site Servers**, select the name of the computer that will run the MSMQ-MQSeries Bridge and then click **Add**.

Add the MSMQ Server to the Foreign Site

The MSMQ Server that is on the same Windows 2000 server as the MSMQ-MQSeries Bridge must be added to the Foreign Site created earlier.

1. From the 'Users and Computers' Snap-in, select **View** and **Advanced Features** followed by **View and Users, Groups and Computers as containers**.
2. Expand the Domain Controllers / <Your server name> / MSMQ folder.
3. Right click the **MSMQ** object and select **Properties**.
4. Select the **Sites** tab.
5. Select the foreign site created earlier, then click **Add** to add this server to the foreign site.

Modify Permissions for the Foreign Site

Perform the following steps to modify permissions for the foreign site.

1. From the 'Sites and Services' Snap-in, select expand the **Sites** folder and select the foreign site created earlier.
2. Select the foreign site, right-click and then select **Properties**.
3. Select the **Security** tab, select **Everyone,** and then enable **Open Connector Queue**.

Fault Tolerance and Load-Balancing

The simplest approach to achieving a fault tolerant and load-balanced MSMQ-MQSeries Bridge environment is to use more than one MSMQ-MQSeries Bridge as part of the same foreign-connected network. Each MSMQ-MQSeries Bridge is configured as described in this document, ensuring that each MSMQ-MQSeries Bridge is part of the **same** foreign-connected network.

Each MSMQ-MQSeries Bridge can be configured to use the following.

- The same MQSeries Queue Manager, for example, when communicating directly with the host.
- Different MQSeries Queue Managers, for example, when each MSMQ-MQSeries Bridge communicates with an MQSeries Queue Manager local to the Bridge (on the same computer).

This approach provides a fault-tolerant environment with the following characteristics.

- **Additional Capacity.** Each MSMQ-MQSeries Bridge is sharing the processing load because MSMQ will distribute MSMQ messages between the connector queues used by each MSMQ-MQSeries Bridge as part of its routing. This approach can also grow to meet increased capacity demands by adding additional MSMQ-MQSeries Bridges.
- **Greater Flexibility.**
    - Any MSMQ 1.0 Server can be used with the MSMQ-MQSeries Bridge because no complexities are associated with clustering an MQIS.
    - It is also possible to use a Microsoft SNA Server on the same computer as the MSMQ-MQSeries Bridge. This is an advantage in a network-bound environment. However, it is still a good idea to use an SNA Client, as well as hot backup and load balancing of SNA LU 6.2 sessions provided by multiple Microsoft SNA Serves in a sub-domain.
- **Reduced Complexity.**
    - It is easier to configure and manage multiple MSMQ-MQSeries Bridges with this approach, rather than using a clustered approach.
    - Software upgrades are simplified when running in a live operational environment.

The following are suggested configurations for using the MSMQ-MQSeries Bridge.

- Windows NT 4.0 Enterprise Edition, MQSeries Client or MQSeries Server, Any MSMQ 1.0 Server and the MSMQ-MQSeries Bridge. Where SNA LU 6.2 is required, use an SNA Client on each Bridge computer.
- Any Windows 2000 Server, MQSeries Client or MQSeries Server version 5.1 and the MSMQ-MQSeries Bridge from SNA Server 4.0 Service Pack 3. Where SNA LU 6.2 is required, use an SNA Client on each Bridge computer.

It is possible to cluster the MSMQ-MQSeries Bridge to provide a fault tolerant environment. However, additional capacity is not provided because the MSMQ-MQSeries Bridge cannot run in an active-active cluster configuration. Software installation and upgrades are also fairly time consuming in these configurations. For these reasons, clustering an MSMQ-MQSeries Bridge is not recommended.

Programming Topics

The MSMQ-MQSeries Bridge performs message property conversions when routing messages between MSMQ and MQSeries. These are documented in the MSMQ-MQSeries Bridge Online Guide.

This section highlights some of those message property conversions, and additional guidelines for programming an application that uses the MSMQ-MQSeries Bridge. This section assumes a high-level of knowledge of both the MSMQ COM API, C API and MSMQ message properties.

Overriding Bridge Conversions

The MSMQ-MQSeries Online guide provides good information regarding which MSMQ message properties are converted to which MQSeries message properties, as well as information on message properties not converted. Depending on the MQSeries application with which integration is being performed, it might become necessary to override the conversions performed by the MSMQ-MQSeries Bridge.

Overriding the MSMQ-MQSeries Bridge conversions focuses around an MQMD. An MQ Message Descriptor (MQMD) is a message header that contains information such as message type, message expiry time, and which MQSeries Queue Manager sent the message.

At the programming level, an MQMD is a C data structure containing fields that equate to many MSMQ message properties. For example, the MQMD Format field equates to the MSMQ message body type (PROPID_M_BODY_TYPE) and the MQMD ReplyToQ field equates to the MSMQ Response Queue (PROPID_M_RESP_QUEUE).

From MSMQ to MQSeries

The MSMQ-MQSeries Bridge provides an API to override the MSMQ-to-MQSeries conversions offered by the Bridge. This API is the Extension Property API. The Extension Property API provides a mechanism to store a complete copy of an MQSeries MQMD data structure in an MSMQ message extension property (PROPID_M_EXTENSION).

When sending a message from MSMQ to MQSeries, the Bridge inspects the extension property of the MSMQ message. If the Bridge finds that the extension property contains an MQMD data structure, then the Bridge uses the values specified in the MQMD to perform its conversions. This provides a mechanism to exchange explicit values of the message fields, supplementing or overriding the default conversions.

Be aware that the MSMQ message extension property is only available through the C API in MSMQ 1.0. This restriction is lifted in MSMQ 2.0.

Note About MQSeries Version 5.0

The MSMQ-MQSeries Bridge is capable of working with MQSeries version 5. However, when using the Extension Property API to override Bridge conversions in conjunction with MQSeries version 5, the conversion fails and the MSMQ-MQSeries Bridge places the MSMQ message in one of the mqbridge dead-letter queues. SNA Server 4.0 Service Pack 3 addresses this issue. A separate hot fix (QFE 4122) also addresses this issue.

MQ Message Descriptor ReplyToQ Field

The MSMQ-MQSeries Bridge will set the ReplyToQ field of an MQMD when a message is sent from MSMQ to MQSeries through the Bridge. The receiving application can then use that value as the destination address for application-response messages.

By default, the MSMQ-MQSeries Bridge places the Windows NT computer name into the ReplyToQ field of an MQMD. If it is required to re-direct messages to a different MSMQ-MQSeries Bridge (such as application-response messages, report messages and acknowledgement messages), then the ReplyToQ field must be overridden.

There are two ways to do this.

- Within the MSMQ-MQSeries Explorer, right-click the connected network and select **Properties** from the menu. The Reply To QM Name appears on the **General** tab. Changing this value will change the value placed in the ReplyToQ field of an MQMD.
- Override the ReplyToQ field of an MQMD using the Extension Property API. This approach will override values specified in the MSMQ-MQSeries Explorer. It provides programmatic control over the destination of application-response messages, report messages and acknowledgement messages.

From MQSeries to MSMQ

No automatic override of the Bridge conversions occurs in the MQSeries-to-MSMQ direction. When an MQSeries message passes through the Bridge, the Bridge converts the MQSeries message to an MSMQ message. It then populates the MSMQ Extension Property with the MQMD data structure from the converted MQSeries message.

Therefore, it is possible for an MSMQ application to access the MSMQ Extension Property to obtain the original values contained within the MQMD data structure before the Bridge converted the MQSeries message.

Sample Program

The MSMQ-MQSeries Bridge provides a sample program that demonstrates how to use the Extension Property API. This program also provides an MQMD Helper COM Component that makes it easier to manipulate an MQMD.

The MQMD Helper component provides a COM interface that enables a client to specify values for a number of fields on an MQMD. The MQMD Helper component uses the Extension Property API, and uses the values supplied by the client to construct and return a byte array that contains an appropriately populated

MQMD data structure. The client can then use the returned byte array to populate the MSMQ message extension property, overriding the Bridge conversions.

The MQMD Helper component also enables a client to extract a number of fields from an MQMD. The client supplies the MSMQ message extension property (as a byte array). The MQMD Helper component decodes the byte array into an MQMD using the Extension Property API.

Message Persistence

From MSMQ to MQSeries

MSMQ express messages are not stored as persistent data to disk. If the MSMQ-MQSeries Bridge is not running and an MSMQ express message is sent through the Bridge, it will thus be placed in an internal connector queue and will remain there. If the server that houses the Bridge is re-booted before the message is processed, then the message is lost. This is normal behavior for MSMQ express messages and enables application developers to balance performance with robustness. Recoverable and transactional messages survive server re-boots and are stored in files under the `MSMQ\Storage\LQS` directory.

From MQSeries to MSMQ

MQSeries messages are always stored as persistent data to disk and therefore survive re-boots. If the MSMQ-MQSeries Bridge is not running, and an MQSeries message is sent through the Bridge, it will thus be placed in a transmission queue. The MQSeries message will be stored in a file under the `<IBM MQSeries Install Point>\qmgrs\<Q Mgr Name>\Queues\<Transmission Queue Name>` directory.

Transactions through the Bridge

MSMQ and MQSeries applications that send messages in a transaction through the MSMQ-MQSeries Bridge can expect the same semantics and notifications that would be received if the application were sending messages to native queues. When an MSMQ application commits to sending a message, the message is first placed in the source xact dead-letter queue, and then routed to the transactional connector queue for processing by the Bridge. The Bridge will read the message within a transaction; it will also convert and send the MQSeries message within a transaction. Because MQSeries only supports a single expiration field, the smaller of the two MSMQ timeout values PROPID_M_TIME_TO_REACH_QUEUE and `PROPID_M_TIME_TO_BE_RECEIVED` is used to set the MQSeries Expiry value.

Acknowledgement messages (reports) sent from MQSeries are routed to the transmission queues used by the MSMQ-MQSeries Bridge. The Bridge reads and converts the acknowledgement messages to equivalent MSMQ acknowledgements. It then sends the acknowledgement to the MSMQ queue specified by the sending application.

Similar processing occurs in the MQSeries-to-MSMQ direction. The MQSeries Expiry value is converted to the MSMQ `PROPID_M_TIME_TO_BE_RECEIVED` timeout value. The MSMQ timeout value `PROPID_M_TIME_TO_REACH_QUEUE` is not set.

Unicode, ASCII, EBCDIC and MSMQ Body Types

The MSMQ-MQSeries Bridge only performs conversion of message bodies where an MSMQ message contains a Unicode string and where an MQSeries message contains an ASCII string.

From MSMQ to MQSeries

The MSMQ-MQSeries Bridge will handle the conversion of MSMQ message bodies from Unicode to ASCII, but only when the MSMQ message body type (PROPID_M_BODY_TYPE) is VT_BSTR. In this case, the Bridge will create an MQSeries message with a format (MQMD.Format) set to MQFMT_STRING.

As an aside, the MSMQMessage COM object will take care of setting the MSMQ message body type to VT_BSTR when a Visual Basic string (BSTR) is assigned to the MSMQMessage.Body property.

The MSMQ-MQSeries Bridge does not handle conversion from ASCII to EBCDIC. If interfacing with an MQSeries application that requires an EBCDIC message body, then perform one of the following:

1) Route the MQSeries message through an ASCII-to-EBCDIC converter before routing the MQSeries message to the actual application; or,

2) Perform application-level ASCII-to-EBCDIC conversion and write the resulting byte array to the MSMQMessage.Body property before sending the message through the Bridge. Because the message body does not contain a string, the MSMQMessage COM object will not set the body type to VT_BSTR (it is actually set to VT_UI1 | VT_ARRAY) and the Bridge will not perform any conversions.

Be aware that the MSMQ message body type can only be manipulated through the MSMQ C API, and not through the MSMQ COM API. This applies to both MSMQ 1.0 and MSMQ 2.0.

From MQSeries to MSMQ

The MSMQ-MQSeries Bridge will handle the conversion of MQSeries message bodies from ASCII to Unicode but only when the MQSeries message format (MQMD.Format) is MQFMT_STRING. In this case, the Bridge will create an MSMQ message with a body type set to VT_BSTR, and it will be possible to assign the MSMQMessage.Body property to a Visual Basic string.

Again, the MSMQ-MQSeries Bridge does not handle conversion from EBCDIC to ASCII. Similar suggestions apply as were given for the MSMQ-to-MQSeries direction.

If the MQSeries message format is not MQFMT_STRING, the Bridge will not perform any conversions. The resulting MSMQ message will have a body type set to VT_NULL. Unfortunately, this precludes reading the MSMQ message body using the MSMQ COM API. The solution is to use the MSMQ C API and treat the MSMQ message body as a byte array (VT_UI1 | VT_ARRAY).

MSMQ Delivery Type and Journal Message Properties

An MSMQ message sent with the following combination will result in the value MQRO_DISCARD assigned to the Report field of the MQMD data structure.

MSMQ Message Property	**MSMQ Message Property Value**
PROPID_M_DELIVERY_TYPE	MQMSG_DELIVERY_EXPRESS
PROPID_M_JOURNAL	MQMSG_JOURNAL_NONE

The value of MQRO_DISCARD results in MQSeries discarding messages. The values for the MSMQ Message Properties listed above are the default values that MSMQ assumes for messages.

It is not possible to override this Bridge conversion using the Extension Property API. To avoid the loss of MQSeries messages use one of the following combinations.

MSMQ Message Property	**MSMQ Message Property Value**
PROPID_M_DELIVERY_TYPE	MQMSG_DELIVERY_EXPRESS
PROPID_M_JOURNAL	MQMSG_JOURNAL
OR	
PROPID_M_DELIVERY_TYPE	MQMSG_DELIVERY_EXPRESS
PROPID_M_JOURNAL	MQMSG_JOURNAL_DEAD_LETTER
OR	
PROPID_M_DELIVERY_TYPE	MQMSG_DELIVERY_RECOVERABLE
PROPID_M_JOURNAL	MQMSG_JOURNAL_NONE

These MSMQ Message properties are accessible through the MSMQ COM API and equate to MSMQMessage.Delivery and MSMQMessage.Journal.

Additional Information

MQSeries Definitions Explained

The following definitions were obtained using the export facility of the MSMQ-MQSeries Bridge Explorer. These show the following.

- The channel definition used for communication between the MQSeries Client (Bridge) and an MQSeries Queue Manager (MQQM).
- The transmission queues used for sending transactional and non-transactional messages from MQSeries to MSMQ.
- The queue-manager aliases used to associate the Bridge with the transmission queues.

These definitions are imported into the MQSeries Queue Manager as part of the Bridge configuration.

Client-Definition File

```
*************  Define MQI client side channels *************

* Define TCP/IP channel 'MQQM.CLIENT.CHL'.
        DEFINE  CHANNEL(MQQM.CLIENT.CHL) +
                CHLTYPE(CLNTCONN) +
                DESCR('MQI channel') +
                TRPTYPE(TCP) +
                CONNAME('10.10.10.10’(1414)') +
                QMNAME(MQQM) +
                REPLACE
```

This defines the client side of an MQI Channel for communication between an MQSeries Client and an MQSeries Queue Manager that is named MQQM. TCP is the protocol used for communication between the Client and Queue Manager. The Queue Manager is listening on port 1414 at address 10.10.10.10.

Server-Definition File

```
*************  Define MQI server side channels *************

* Define TCP/IP channel 'MQQM.CLIENT.CHL'.
        DEFINE  CHANNEL(MQQM.CLIENT.CHL) +
                CHLTYPE(SVRCONN) +
                DESCR('MQI channel') +
                TRPTYPE(TCP) +
                MCAUSER(' ')
```

This defines the server-side of an MQI Channel for communication between an MQSeries Client and an MQSeries Queue Manager.

```
************  Define model queue for the SyncQ  ************

        DEFINE  QMODEL(Q2Q_SYNC_Q) REPLACE +
                DESCR('Microsoft MSMQ-MQSeries Bridge SyncQ') +
                DEFTYPE(PERMDYN)
```

This defines an MQSeries model queue. A model queue is not a real queue but a template that is used when creating dynamic queues with the MQSeries MQOpen API call.

The MSMQ-MQSeries Bridge dynamically creates an MQSeries sync queue when sending transactional messages from MSMQ to MQSeries. The sync queue is used as a ‘scratch pad’ to prevent duplicate sending of MSMQ messages to MQSeries queues when part of a transaction.

One sync queue is created and used by each MSMQ-MQSeries Bridge for all transactional messages that are sent from MSMQ to MQSeries. A similar sync queue is used in the MQSeries-to-MSMQ direction but the sync queue is an MSMQ queue managed by the MSMQ-MQSeries Bridge.

```
***********  Define MQSeries transmission queues ***********

* Define Normal service transmission queue.
        DEFINE  QLOCAL(MQQM.CLIENT.CHL.XMITQ) REPLACE +
                USAGE(XMITQ) +
                DESCR('XmitQ for MQQM.CLIENT.CHL Normal')

* Define High service transmission queue.
        DEFINE  QLOCAL(MQQM.CLIENT.CHL.XMITQ.HIGH) REPLACE +
                USAGE(XMITQ) +
                DESCR('XmitQ for MQQM.CLIENT.CHL High')
```

This defines two transmission queues that are used by MQSeries to deliver messages to the Bridge. In the MQSeries-to-MSMQ direction, transactional messages are sent using normal service, and non-transactional messages are sent using high service.

```
***  Define alias queue manager for the bridge  computer  ***

* Normal service.
        DEFINE  QREMOTE(BRIDGE1) REPLACE +
                DESCR('QM Alias') +
                RQMNAME('BRIDGE1') +
                XMITQ(MQQM.CLIENT.CHL.XMITQ)

* High service.
        DEFINE  QREMOTE(BRIDGE1%) REPLACE +
                DESCR('QM Alias') +
                RQMNAME('BRIDGE1%') +
                XMITQ(MQQM.CLIENT.CHL.XMITQ.HIGH)
```

MQSeries requires an alias that maps a transmission queue to a particular MSMQ computer. This allows an MQ Queue Manager to work out the transmission queue on which it should place a message that has been received and is destined for another Queue Manager.

In this example, an MQSeries Queue Manager will place an MQSeries message destined for an MSMQ Queue on the MSMQ computer (BRIDGE1) onto one of these two transmission queues. The actual transmission queue used depends on whether the message was sent as normal or high service.

The Bridge then reads MQSeries messages from these transmission queues, converts the message and delivers the MSMQ message to the appropriate MSMQ queue.

Configuring MQ Series to Send Messages to Specific MSMQ Queues

It is also possible to expose other MSMQ computers and actual MSMQ Queues to the MQSeries environment by using additional server aliases. These MSMQ computers and MSMQ queues need not be physically on the actual Bridge computer. The additional aliases would instruct an MQSeries Queue Manager to route messages destined for other MSMQ computers to the same MQSeries transmission queues that the Bridge is already using. The Bridge will then take care of reading, converting and routing the message to its final destination.

By defining MSMQ Computers and Queues within the MQSeries environment, it is then possible for MQSeries applications to address messages to specific MSMQ Queues.

MQSeries only requires queue-manager aliases to be defined to route messages between MQ Queue Managers. The Bridge behaves as an MQ Queue Manager. Exposing specific MSMQ Queues to the MQSeries environment reduces the likelihood of errors in the MQSeries configuration.

In this example below, 'TEST' is the name of the MSMQ Computer exposed to MQSeries.

```
***  Define alias queue manager for any specific  computer  ***
*
* Normal service.
*       DEFINE  QREMOTE(TEST) REPLACE +
*               DESCR('QM Alias') +
*               RQMNAME(TEST) +
*               XMITQ('MQQM.CLIENT.CHL.XMITQ')

* High service.
*       DEFINE  QREMOTE(TEST%) REPLACE +
*               DESCR('QM Alias') +
*               RQMNAME(TEST%) +
*               XMITQ('MQQM.CLIENT.CHL.XMITQ.HIGH')
```

In this example below, QUEUE4 on MSMQ Computer MACHINE2 is exposed to MQSeries.

```
*Normal Service
DEFINE QREMOTE(QUEUE4) +
        DESCR('Queue Alias') +
        RNAME(QUEUE4)
        RQMNAME(MACHINE2) +
        XMITQ('MQQM.CLIENT.CHL.XMITQ')

*High Service
DEFINE QREMOTE(QUEUE4%) +
        DESCR('Queue Alias') +
        RNAME(QUEUE4)
        RQMNAME(MACHINE2%) +
          XMITQ('MQQM.CLIENT.CHL.XMITQ.HIGH')
```

MSMQ-MQSeries Bridge Queues

The MSMQ-MQSeries Bridge specifically uses a number of MSMQ queues as follows.

Public Queues

mqbridge dead letter	Messages sent from MQSeries to MSMQ using MQSeries High Service arrive here if, for example, the message is addressed to a non-existent queue or a timer expires.
mqbridge xact dead letter	Messages sent from MQSeries to MSMQ using MQSeries normal service arrive here if, for example, the message is addressed to a non-existent queue or a timer expires.
mqbridge proxy controller	Used to provide remote management of the MSMQ-MQSeries Bridge NT Service.

Private Queues

q2qchannel controlxxx	Where XXX is a number. Four of these exist per connected network, one for each message pipe. These queues provide remote management of each message pipe.
q2q sync queue.<Chl>.sync	Where <Chl> is the name of the MQI Client Channel. This queue is used to prevent duplicate messages being sent from MQSeries to MSMQ when part of a transaction. This queue provides similar functionality as the MQSeries model queue.

Limitations

Encryption

Currently, the MSMQ-MQSeries Bridge does not provide support for MSMQ encryption. Encrypted MSMQ messages cannot be submitted to the MSMQ-MQSeries Bridge. This restriction will be lifted in the next version of the MSMQ-MQSeries Bridge.

MSMQ Message Body Size

MSMQ Messages are limited to 4 MB of data within the message body. Where a message body is populated with a Unicode string, the actual message body size becomes twice the length of the actual Unicode string. This can have implications for large message sizes sent through the MSMQ-MQSeries Bridge.

In the MSMQ-to-MQSeries direction, the MSMQ-MQSeries Bridge will convert string-message bodies (PROPID_M_BODY_TYPE = VT_BSTR) from Unicode to ASCII. A Unicode string 100 characters long thus results in a message body size of 200 bytes; but when passed through the Bridge, it results in a message body size of 100 bytes and an ASCII string 100 characters long.

However, in the MQSeries-to-MSMQ direction, the MSMQ-MQSeries Bridge will convert string-message bodies (MQMD.Format = MQFMT_STRING) from ASCII to Unicode. When passed through the Bridge, an ASCII string 100 characters long results in a message body size of 200 bytes, and a Unicode string 100 characters long. If the actual string-message body is approaching 2 MB in size, this will result in an MSMQ message body that is larger than permitted. In this scenario, the MQSeries message does not pass through the Bridge, generating an NT event log message.

This restriction will be lifted in the next version of the MSMQ-MQSeries Bridge, but only in the MQSeries-to-MSMQ direction.

MSMQ Message Body Type

With the MSMQ (1.0 and 2.0) COM API, it is not possible to control the value of PROPID_M_BODY_TYPE. The MSMQ COM API will automatically set the value of this MSMQ message property based on what is actually assigned to the MSMQ message body.

The MSMQ C-API must be used to control the message-body type specified for the MSMQ message.

Dead-Letter Queue Notifications

If an MQSeries message is sent to MSMQ and addressed to a non-existent MSMQ Queue, the MSMQ-MQSeries Bridge will place the message in one of the Bridge dead-letter queues. MQSeries is not notified that the message could not be delivered. This is because MQSeries only requires queue-manager aliases to be defined to route messages between MQ Queue Managers.

An MSMQ message cannot be sent to MQSeries without the MQSeries Queue exposed as an MSMQ foreign queue within the MSMQ Explorer. However, it is still possible that the MQSeries queue has not actually been defined within MQSeries. In this scenario, MSMQ messages sent through the Bridge will appear in an MQSeries dead-letter queue and MSMQ is not notified that the message could not be delivered.

The lack of notifications in these scenarios should not be an issue in a properly configured MQSeries and MSMQ environment.

References

MSMQ-MQSeries On-line Guide

MSMQ Administrators Guide

IBM MQSeries Administrators Guide

◆ ◆ ◆ ◆