

MQSeries



Link for R/3 User's Guide

Version 1.2

MQSeries



Link for R/3 User's Guide

Version 1.2

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix D. Notices" on page 125.

Fifth edition (February 2001)

This edition applies to the following products:

- IBM MQSeries link for R/3 Version 1.2 for AIX, program number 5765-B66
- IBM MQSeries link for R/3 Version 1.2 for AS/400, program number 5733-A22
- IBM MQSeries link for R/3 Version 1.2 for Compaq Tru64 UNIX, program number 5648-D57
- IBM MQSeries link for R/3 Version 1.2 for HP-UX, program number 5765-C01
- IBM MQSeries link for R/3 Version 1.2 for OS/390, program number 5655-F39
- IBM MQSeries link for R/3 Version 1.2 for Sun Solaris, program number 5765-B98
- IBM MQSeries link for R/3 Version 1.2 for Windows NT, program number 5765-B99

and to any subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright International Business Machines Corporation 1997, 2001. All rights reserved.**

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

Contents

Figures	vii	Chapter 4. Installing MQSeries link for R/3	15
Tables	ix	READ.ME file	15
About this book	xi	Installing MQSeries link for R/3 on AIX	
Who this book is for	xi	Version 4.1	15
What you need to know to understand this		Installing MQSeries link for R/3 on Compaq	
book.	xi	Tru64 UNIX Version 4.0F or 5.0	16
Terms used in this book	xii	Installing MQSeries link for R/3 on HP-UX	
How to use this book	xii	Version 10.	16
Summary of changes	xiii	Installing MQSeries link for R/3 on Sun	
Changes in this edition (GC33-1934-04).	xiii	Solaris Version 2.5	17
Changes for the fourth edition		Installing MQSeries link for R/3 on OS/390	
(GC34-1934-03)	xiii	Version 5.2	17
Changes in the third edition (GC33-1934-02)	xiv	Installing MQSeries link for R/3 on OS/400	
		Version 4	17
Chapter 1. Introduction to MQSeries link for		Installing MQSeries link for R/3 on Windows	
R/3	1	NT	18
Overview	1	Installing MQSeries link for R/3 on	
The benefits of using MQSeries link for R/3	2	Windows NT Version 3.5.	18
Understanding MQSeries link for R/3 servers	3	Installing MQSeries link for R/3 on	
Outbound servers	3	Windows NT Version 4.0.	18
Inbound servers	6	Directories after installation	19
Understanding user exits	8	Uninstalling MQSeries link for R/3 from AIX	20
An example of when to use exits	9	Uninstalling MQSeries link for R/3 from	
Summary of the MQSeries link for R/3		Compaq Tru64 UNIX	20
components	10	Uninstalling MQSeries link for R/3 from	
		HP-UX.	20
Chapter 2. Planning for MQSeries link for		Uninstalling MQSeries link for R/3 from Sun	
R/3	11	Solaris	20
Hardware requirements	11	Uninstalling MQSeries link for R/3 from	
Software requirements	11	OS/390	21
Compaq Tru64 UNIX	11	Uninstalling MQSeries link for R/3 from	
OS/390.	11	OS/400	21
Windows NT.	11	Uninstalling MQSeries link for R/3 from	
Restrictions	12	Windows NT.	21
Supported platforms	12	Removing MQSeries link for R/3 from	
MQSeries platforms	12	Windows NT Version 3.5.	21
		Removing MQSeries link for R/3 from	
Chapter 3. Migrating from version 1.0 to		Windows NT Version 4.0.	21
version 1.2	13		
Migration procedure	13	Chapter 5. Configuring MQSeries link for	
		R/3	23
		Understanding which MQSeries objects you	
		need	23
		Defining a queue manager	24
		Data conversion exit library.	24

Defining MQSeries objects for an outbound server	24	The Initialise entry point (smqUserExitInitialise)	49
Defining MQSeries objects for an inbound server	25	Execute entry point (smqUserExitExecute)	50
Using the Sample MQSC command file	26	Return entry point (smqUserExitReturn)	52
Example of an MQSeries configuration	26	Terminate entry point (smqUserExitTerminate)	52
Advanced queue manager configuration	28	Sample user exits	53
Outbound servers	28	Compiling user exits	53
Inbound servers.	28	Message formats	55
Understanding RFC destinations	29	MQSeries link for R/3 header structure	56
Specifying a remote R/3 client	29	Constructing messages with IDocs	58
Configuring MQSeries link for R/3 for OS/390	30	Chapter 7. Command reference	59
Accessing the OS/390 shell	30	Initialization parameters	59
Starting the MQSeries queue manager	30	smqsi (Start inbound server)	60
Configuration tasks	31	smqso (Start outbound server)	62
Task 1. Defining TCP/IP Ports for use with the operating system	31	Chapter 8. Running servers as Windows NT services	67
Task 2. Creating the MQSeries objects	31	Changes to the Windows NT system registry	68
Task 3. Defining the RFC destinations on R/3	32	Removing the Out and In Bound Servers from the Service Control Manager	69
Task 4. Setup for inbound server load balancing	34	Chapter 9. Initialization files	71
Task 4a. Creating a SAP logon group.	34	Outbound server initialization parameters	71
Task 4b. Mapping inbound server to SAP logon group	35	RFC destination parameters.	71
Task 5. Mapping R/3 Logical Systems to MQSeries destinations	35	Outbound user exit defaults.	72
Configuration parameters in the smqDestConf file	36	MQSeries definitions	73
Task 6. Specifying server configuration in the initialization (ini) files.	39	Other defaults for the outbound server	73
Initializing the outbound server	39	Inbound server initialization parameters	75
Initializing the inbound server	39	R/3 system connection defaults	75
Task 7. Starting the servers	40	MQSeries definitions used by the inbound server	76
Creating the MQSeries environment for OS/390	40	User exit information	78
The start commands	40	Other defaults for the inbound server	79
Starting multiple instances of the outbound server.	41	saprfc.ini file information	80
Testing the basic configuration	41	Chapter 10. Troubleshooting	83
1. Starting the outbound server	42	Outbound server problem diagnosis	83
2. Starting the inbound server	42	Starting the outbound server on OS/390	84
3. Sending the IDocs	42	Inbound server problem diagnosis	84
4. Monitoring IDoc traffic	43	Data Conversion problems	85
Chapter 6. Writing user exits.	45	Communication and system failures	85
About user exits	45	What happens when MQSeries link for R/3 detects a failure	85
When the exits are called	45	Handling unrecognized messages	86
Enabling the user exits	47	Contents of a bad message	87
Understanding the external interface.	48	Bad message diagnosis	87
		Reprocessing bad messages	89

Using trace and error logging to diagnose problems	90
MQSeries link for R/3 trace.	90
MQSeries link for R/3 Error logging	90
Error logging on OS/390.	91
R/3 Error Logging	92
R/3 Remote function call (RFC) trace	92
Chapter 11. Security	93
Security precautions	93
Appendix A. Samples	95
Sample initialization files	95
Sample configuration file.	97
Sample user exits	99
Sample data conversion exit	100
Sample exit for distributed platforms	101
Sample data conversion exit for OS/390	101
MQSC command file (smqscdef.tst)	102
OS/390 samples	103
Appendix B. Quick reference	105
Outbound configuration	105
Inbound configuration	106

Appendix C. Messages and codes	109
Appendix D. Notices	125
Trademarks	127
Glossary of terms and abbreviations	129
Bibliography	131
MQSeries cross-platform publications	131
MQSeries platform-specific publications	131
Softcopy books.	133
HTML format	133
Portable Document Format (PDF)	134
BookManager [®] format	134
PostScript format	134
Windows Help format	134
MQSeries information available on the	
Internet	134
Related publications	134
Index	135
Sending your comments to IBM	139

Figures

1.	MQSeries link for R/3	1	10.	Entry points in a user exit	49
2.	MQSeries link for R/3 overview	2	11.	MQSeries message structure and IDocs	55
3.	Outbound and Inbound servers	3	12.	Structure of the MQSeries link for R/3 header	57
4.	Outbound transactions	4	13.	Structure of the bad message header.	87
5.	Inbound transactions	6	14.	Sample outbound ini file showing sample parameters	95
6.	Exit handlers and user exits on the outbound server	8	15.	Sample inbound ini file showing sample parameters	96
7.	MQSeries link for R/3 and MQSeries	27	16.	sample rfc.ini file	96
8.	Data flow through the outbound server user exit	46	17.	Sample smqDestConf file	98
9.	Data flow through the inbound server user exit after initialization	47			

Tables

1.	Error types	88	3.	Outbound configuration parameters	105
2.	Reason codes	88	4.	Inbound configuration parameters	106

About this book

The *MQSeries Link for R/3 User's Guide* describes the MQSeries link for R/3 product: what it is, how to install it on your system, and how you can use it.

This book includes these chapters:

- “Changes in this edition (GC33-1934-04)” on page xiii
- “Chapter 1. Introduction to MQSeries link for R/3” on page 1
- “Chapter 2. Planning for MQSeries link for R/3” on page 11
- “Chapter 3. Migrating from version 1.0 to version 1.2” on page 13
- “Chapter 4. Installing MQSeries link for R/3” on page 15
- “Chapter 5. Configuring MQSeries link for R/3” on page 23
- “Chapter 6. Writing user exits” on page 45
- “Chapter 7. Command reference” on page 59
- “Chapter 8. Running servers as Windows NT services” on page 67
- “Chapter 9. Initialization files” on page 71
- “Chapter 10. Troubleshooting” on page 83
- “Chapter 11. Security” on page 93
- “Appendix A. Samples” on page 95
- “Appendix B. Quick reference” on page 105
- “Appendix C. Messages and codes” on page 109
- “Glossary of terms and abbreviations” on page 129

Who this book is for

This User's Guide is written for anyone who runs business applications that use R/3 and who needs to implement commercial messaging using the MQSeries family of products. This book will be of particular interest if you intend to design, program, implement, administer, maintain, or support systems that use the MQSeries link for R/3.

To use this book, you should have a general understanding of R/3 application systems together with some experience or knowledge of messaging using the MQSeries family of products.

What you need to know to understand this book

To use MQSeries link for R/3, you should have a good understanding of SAP, especially the Application Enabling Layer (ALE) and IDoc formats. Refer to your SAP documentation for a complete description of ALE and IDocs.

About this book

Terms used in this book

The terms *outbound* and *inbound* are used with reference to SAP. A message will therefore be “outbound from SAP” or “inbound to SAP”.

How to use this book

This book tells you about preparing for MQSeries link for R/3, and includes full descriptions of installation and configuration. It also contains guidance about writing user exits and reference information about running the servers. This book also helps you with problem solving, including a complete list of messages and codes. It describes the samples provided with the product and there is a glossary and a bibliography at the back of the book.

Summary of changes

This section describes changes in this edition of *MQSeries link for R/3 User's Guide*. Changes since the previous edition of the book are marked by vertical lines to the left of the changes.

Changes in this edition (GC33-1934-04)

Version 1.2 now includes support for:

- MQSeries link for R/3 on Compaq Tru64 UNIX
- MQSeries link for R/3 on OS/390

Additional information about this support is in the following chapters:

- “Chapter 2. Planning for MQSeries link for R/3” on page 11
- “Chapter 4. Installing MQSeries link for R/3” on page 15
- “Chapter 5. Configuring MQSeries link for R/3” on page 23
- “Chapter 6. Writing user exits” on page 45
- “Chapter 7. Command reference” on page 59
- “Chapter 10. Troubleshooting” on page 83
- “Appendix A. Samples” on page 95

Changes for the fourth edition (GC34-1934-03)

The major changes introduced in Version 1.2 of MQSeries link for R/3 are:

Support for SAP R/3 Versions 3 and 4 IDocs

The SAP R/3 Version 4 IDoc structures EDI_DC40 and EDI_DD40 and the existing Version 3 structures are all supported by this version of MQSeries link for R/3.

Load balancing support on inbound server

The inbound server now supports SAP load balancing. By using logon groups, R/3 system administrators can balance the inbound server traffic over multiple application servers.

Running multiple outbound servers to the same program id

You can now start multiple instances of the outbound server to help speed the processing of IDocs.

Sample data conversion exit added

Data conversion may be required when passing data between non-compatible systems.

Changes

Outbound support for batched Idocs

A batch of Idocs can be placed into a single MQ message on the outbound server.

Configuration

This version is configured by editing the smqDestConf file and the 'MQS Options panel' on the SAP system is no longer used.

Changes in the third edition (GC33-1934-02)

The major changes introduced in Version 1.1 of MQSeries link for R/3 affect the outbound server configuration, giving the user more flexibility in R/3 configurations.

Multiple MQSeries destinations

The outbound server now determines the MQSeries destination depending on the receiving logical system and the IDoc type found in the control block of an IDoc. This means that messages sent to the same logical system but of different IDoc types can be routed to different MQSeries destinations.

Improved performance

The outbound server now reads all the required configuration information at start up. This improves performance as RFC calls are no longer needed to query the RFC destination for the MQSeries Queueing Options.

Increased Message size

The maximum message size has been increased to 100MB when used with MQSeries Version 5 servers.

Chapter 1. Introduction to MQSeries link for R/3

This chapter gives you an overview of the MQSeries link for R/3. It gives you a general description of the product, including the MQSeries link for R/3 servers, and the user exits.

Overview

MQSeries link for R/3 is an interface that enables you to integrate your R/3 applications with applications running in other environments, including those on SAP R/3 (referred to in this book as the R/3 system) and R/2 systems.

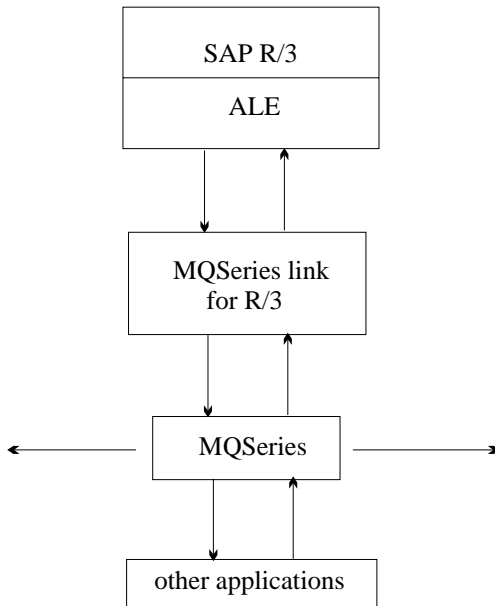


Figure 1. MQSeries link for R/3. MQSeries link for R/3 connects R/3 systems to other applications that use MQSeries messaging.

MQSeries link for R/3 works with the Application Link Enabling (ALE) layer of the R/3 system to transmit Intermediate Documents (IDocs) into and out of your R/3 system, using MQSeries messages and queues to carry the information. It extends the scope of your business by allowing you to link your R/3 applications to any other application that you can access through MQSeries, even when those applications require different data formats. For more information about converting data between different types of systems, see “Understanding user exits” on page 8.

Introduction

In summary, you can use MQSeries to connect an R/3 system to:

- Other R/3 systems
- R/2 systems
- Any other system that you want to exchange data with an R/3 system, such as a legacy database holding enterprise data that you need to get into R/3

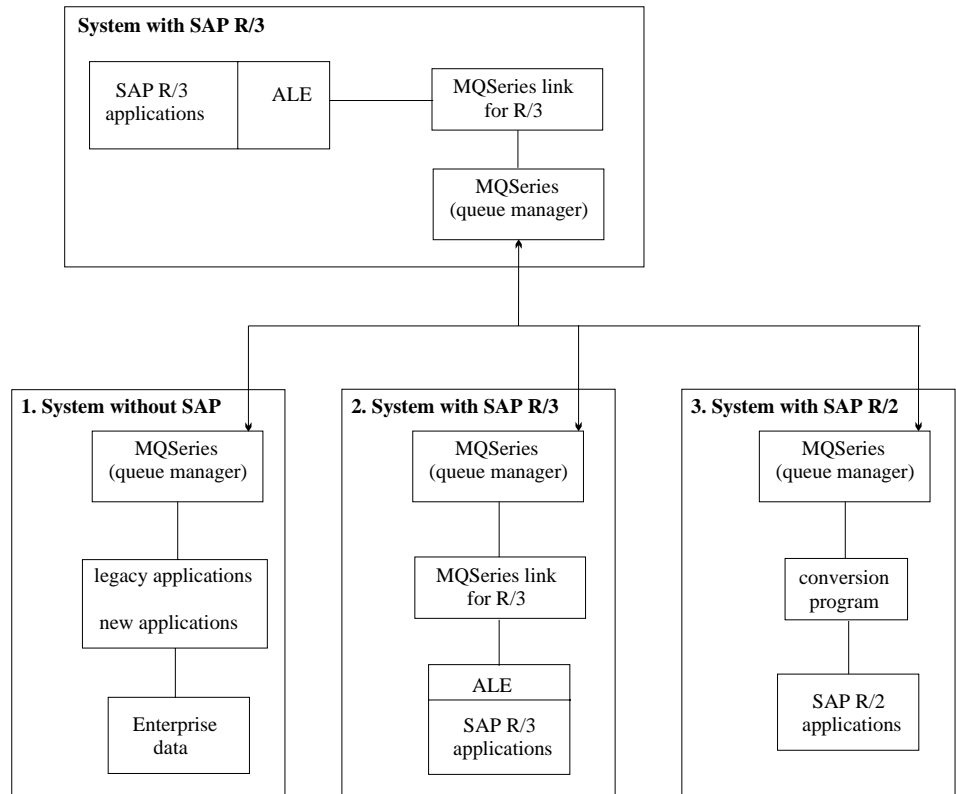


Figure 2. MQSeries link for R/3 overview. MQSeries link for R/3 enables applications running under R/3 to exchange information with (1) Legacy (or new), (2) Other R/3, or (3) SAP R/2 applications.

The benefits of using MQSeries link for R/3

MQSeries link for R/3 provides robust, reliable middleware that connects SAP and other programs across many IBM and non-IBM platforms. It uses programming resources efficiently, makes applications adaptable, and eases network management. It gives programs independence from communications protocols and from the non-availability of the systems, networks, and programs.

With MQSeries link for R/3 you can link your existing R/3 business applications with your other applications, including applications running under older versions of SAP, such as R/2, and non-SAP legacy systems.

Understanding MQSeries link for R/3 servers

There are two main functional components associated with the MQSeries link for R/3 product. These are known as the *outbound server* and the *inbound server*. The outbound server is used when sending data from an R/3 system; the inbound server is used when receiving information into R/3. Both servers have *exit handlers* to give you access to *user exits* outside MQSeries link for R/3 for performing special functions such as data conversion and translation. If you use a user exit, you can either write your own conversion program, using the sample exit `smqsdncv.c` as a template, or use a data translator tool.

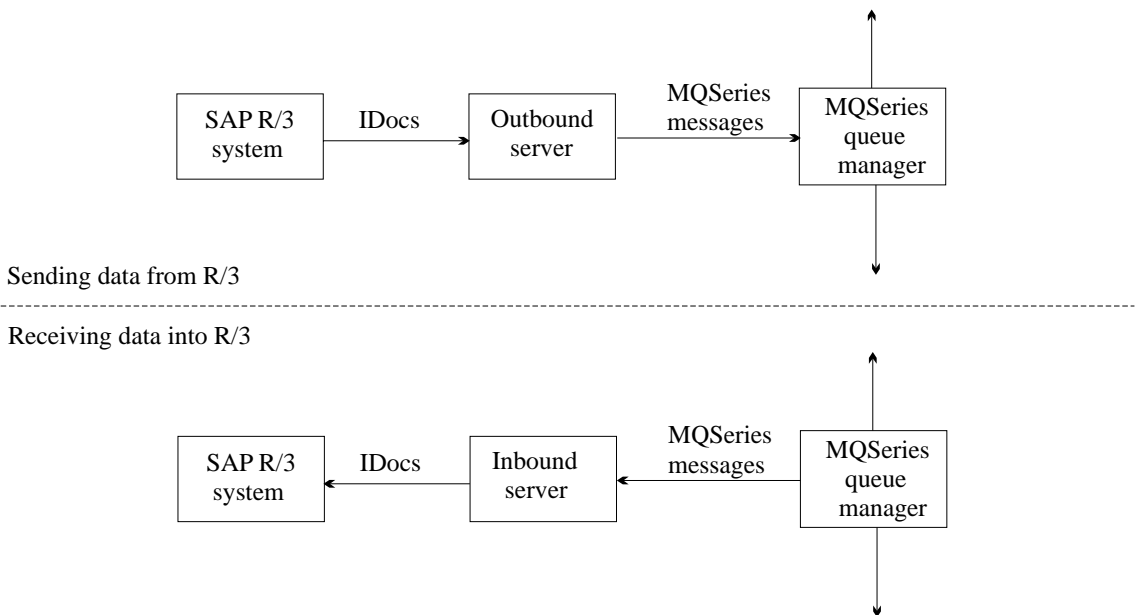


Figure 3. Outbound and Inbound servers

Outbound servers

You start the outbound server using the `smqso` command. Thereafter, the server automatically gets control when an R/3 application sends information through MQSeries. The R/3 application passes a single transaction to MQSeries link for R/3 as a set of one or more IDocs. The outbound server builds the messages and passes them to MQSeries. A configuration file named `smqDestConf` is required to enable the outbound server to determine the

Understanding servers

queues and queue managers to which IDocs should be sent. The server cannot start without this file. (A sample **smqDestConffile** is supplied with MQSeries link for R/3.)

Outbound sequence of events

The following describes the sequence of events when R/3 sends a transaction consisting of one IDoc or multiple IDocs (known as a batch) outbound from an R/3 application (see Figure 4):

1. The outbound server reads the **smqDestConf** file to determine the target queue and queue manager that should be used for each receiving partner and IDoc type.
2. An R/3 application program creates one or more IDocs representing a single transaction, and uses transactional remote function calls (tRFC) to start the outbound process .

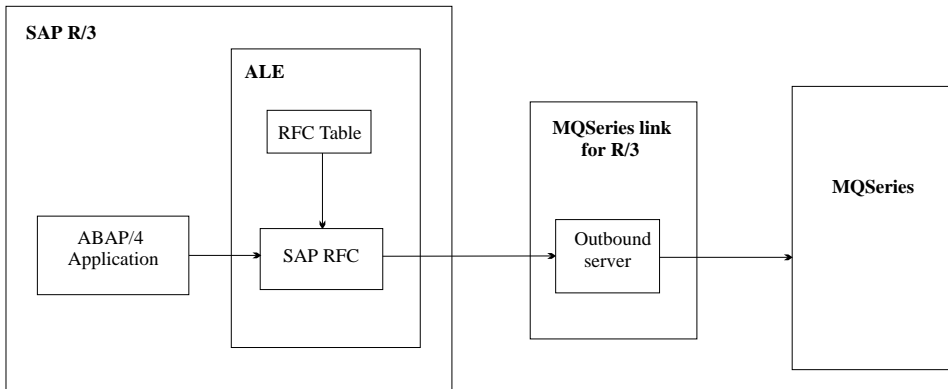


Figure 4. Outbound transactions

3. The RFC component handles the connection to MQSeries link for R/3 and provides the additional control information required to handle the transaction and send it to the specified destination.

The RFC component uses the RFC table (RFCDEST) to identify the connection type and physical destination of the transaction. The table contains entries for all destinations and is maintained using the standard R/3 transaction, sm59.

If the RFC destination program ID matches that of a running outbound server, MQSeries link for R/3 is used.

4. The RFC component passes the transaction data and control information to the MQSeries link for R/3 outbound server. The MQSeries link for R/3 outbound server uses MQSeries destination information (from the control information) and the transaction data to construct the MQSeries messages. Each message contains one or more IDocs for a specific destination.

If the control information indicates that a user exit should be called, the outbound exit handler calls the specified user exit. If the user exit responds 'CONVERSION NOT NEEDED', the message (both descriptor and data) as formatted by MQSeries link for R/3 is sent to MQSeries. If the user exit responds 'CONVERSION SUCCESSFUL', the message (both descriptor and data) as revised by the user exit is sent to MQSeries. If the user exit responds 'CONVERSION FAILED', MQSeries link for R/3 informs the SAPRFC interface of the failure and does not send the message to MQSeries.

The outbound server sends the messages to the specified MQSeries queue. All the messages are put under syncpoint; they are treated as a single logical unit of work.

Batching on the outbound server

SAP supports 'batching' which enables a number of Idocs to be grouped together in a single Message. The MQSeries link for R/3 outbound server supports this function, allowing a number of Idocs to be sent to MQSeries in a single MQSeries message. The *batching* parameter in the outbound *ini* file can be set to determine whether batching should be used.

If you are using batching, all the Idocs in a batch should be of the same type. You should also be aware that batches are not necessarily sent from SAP in the order in which they are created.

See the SAP documentation for information on how to batch IDocs.

Running multiple outbound servers to the same program id

After SAP has sent a batch of IDocs to an outbound server the server will be busy while it is receiving the data. SAP may have more IDocs to send, but these become blocked behind the previous batch. To prevent subsequent batches becoming blocked you can start other instances of the outbound server to deal with them if while the initial server instance is busy. If you want to avoid any blocking of IDoc processing the number of outbound server instances should equal or exceed the maximum number of simultaneous IDoc batches that you expect to receive from SAP.

RFC destinations

Within R/3, you must specify the parameters that uniquely identify the external program, MQSeries link for R/3. MQSeries destinations are type T, that is, TCP/IP. Type T destinations refer to external programs that use R/3 RFC libraries to act as RFC servers. In R/3, you must specify a program ID, which you can supply yourself; when you start an MQSeries link for R/3 server, make sure that you provide this same program ID to the start server command.

Understanding servers

Also, if you are not registering by using the current applications server as your gateway, you must specify the names of an SAP gateway host and service.

SAP provides an R/3 transaction, sm59, for specifying these parameters.

Inbound servers

You start the inbound server using the **smqsi** command. The server gets control whenever an MQSeries message is put on the inbound server queue. The inbound server gets the message from the queue and converts the format if necessary with a user exit, before passing the information in the message as IDocs to the receiving R/3 application.

Note: This means that you have to take steps to ensure that the information is formatted correctly for R/3 processing, or to provide a user exit to convert to the correct format. See “Message formats” on page 55 for more information on message formats, and the sample exit **smqsdcnv.c**

Inbound sequence of events

The following describes the sequence of events when MQSeries link for R/3 receives an inbound transaction from MQSeries for an R/3 application.

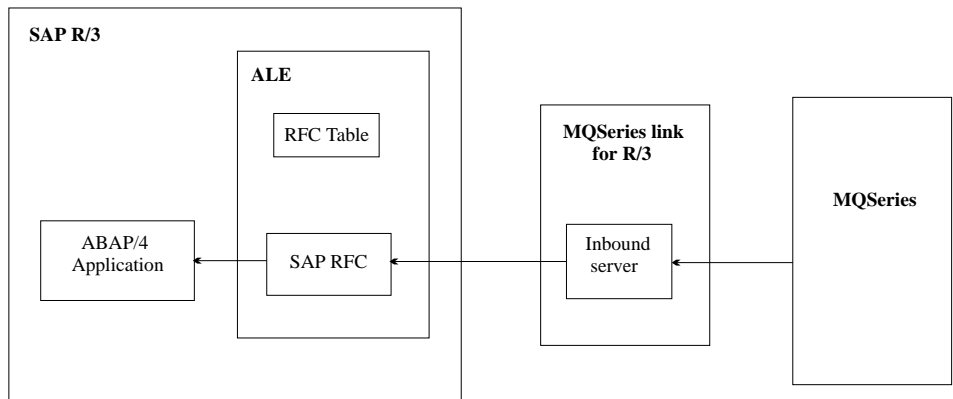


Figure 5. Inbound transactions

1. Once it has been started, the inbound server polls the MQSeries queue for new messages. An application sending information to an R/3 application puts a message on an MQSeries queue. The MQSeries link for R/3 inbound server gets the message from the MQSeries queue, under syncpoint. MQSeries link for R/3 creates IDocs from the incoming transaction data. If the transaction data is to be formatted or converted in some way, MQSeries link for R/3 uses the exit handler to access a user exit written to perform the required processing.

The inbound server processes logs on to SAP as an online dialog user. The link can specify either one specific application server or a logon group. The logon group allows SAP to determine the least loaded application server.

2. The inbound server requests a transaction ID from the R/3 system. MQSeries link for R/3 keeps track of inbound transactions and manages any messages by making an entry in an internal transaction store queue.
3. The link header in the incoming message data contains information about the R/3 application that is to receive the transaction.

If you do not specify the information about the remote R/3 system within the R/3 transaction sm59, the link header outbound message does not contain all the required destination information. In this case, MQSeries link for R/3 uses the default values specified in the initialization (.ini) file for the inbound server of the remote system to make the connection.

The RFC program waits for an inbound message from MQSeries link for R/3. When a message arrives, the RFC program executes the INBOUND_IDOC_PROCESS function for Version 3 Idocs, or the IDOC_INBOUND_ASYNCHRONOUS function for Version 4 Idocs.

4. After the transaction has been transferred successfully to the R/3 system, MQSeries link for R/3 executes an MQSeries commit to delete the inbound message. MQSeries link for R/3 also deletes its entry from the internal transaction store queue.

Inbound load balancing

SAP uses load balancing to distribute users across multiple application servers. SAP considers the capacity of each application server and determines which one is loaded least. The advantages of load balancing are:

- A connection is established to an available application server with the least load
- The connections are independent of specific application servers and are determined at run time

In SAP each user logon equates to a single logon dialog process. The inbound server RFC call logs into SAP as a SAP dialog process. Dialog processes can be load balanced if a logon group is specified at user login. SAP runs the inbound server SAP process on the least loaded application server.

Logon load balancing is only beneficial if there are at least two or more application servers within your configuration that can share the load of the inbound server. You can determine the number of application servers by using the SAP transaction SM51. This transaction also indicates which application server is designated as the 'Message server' with the process ENQUEUE. To ensure the message server is configured properly, check that its hostname is entered in the **hosts** file and that its port number is specified in the **services** file.

Understanding servers

MQSeries allows the user to specify which message server to use by setting the value of the `loadbalance` parameter in the inbound server `ini` file (see “Task 6. Specifying server configuration in the initialization (`ini`) files” on page 39). When this parameter is set to ‘yes’, the hostname value is used to find a matching `DEST` value in the `saprfc.ini` file. The matching `DEST` entry determines which message server should be connected.

Note: The hostname used is the value specified in the MQSeries header file, or if this is blank, the value specified in the inbound `ini` file.

To use load balancing you also need to edit the `saprfc` file “Task 4b. Mapping inbound server to SAP logon group” on page 35 for details.

Understanding user exits

The MQSeries link for R/3 servers provide user exits, which allow you to process messages as they pass through a server. The exits are called by the exit handler associated with each server. Typically, you use the exits to add extra processing functions that are not directly provided by MQSeries link for R/3 or MQSeries, such as reformatting or compressing data in IDocs coming from an R/3 system. You can do these types of operations using your own program or a SAP-approved data translator.

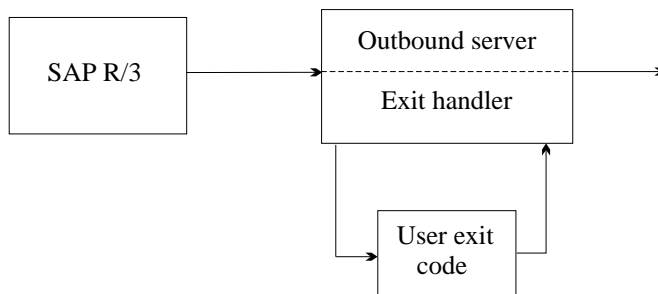


Figure 6. Exit handlers and user exits on the outbound server. You modify the supplied user exit samples to perform any functions that you require.

As with all MQSeries exits, the MQSeries link for R/3 exit handlers execute your user-supplied software in-line, without additional message flows. In MQSeries link for R/3 there are two forms of user exits; an outbound user exit is called before a message is sent to an MQSeries destination queue, an inbound user exit is called after a message has been retrieved from the inbound server message queue.

An example of when to use exits

An R/3 application sends some invoice data to a non-SAP application. The data from the R/3 application is always in IDoc format. However, the receiving application expects the invoice data in a different format.

To overcome this problem, you can write a user exit for the outbound server to:

- Extract the relevant data from the IDoc in the message
- Build a new message with the data in the format expected by the receiving application

The server then puts the new message on its outbound message queue, where MQSeries applications can access it.

For the inbound server, you can use the reverse procedure to convert incoming data from a non-SAP application into an R/3 IDoc format.

Summary of the MQSeries link for R/3 components

Here is a brief overview of some of the components and functions that make up the MQSeries product:

Outbound server

The MQSeries link for R/3 component that accepts IDocs from the source R/3 system and makes them available to other applications connected by MQSeries.

Inbound server

The MQSeries link for R/3 component that receives information from applications connected by MQSeries and passes the information to the destination R/3 system.

C source header file

A header file containing structure definitions required to process MQSeries link for R/3 message data. This is useful for writing your own user exits.

Sample user exits

C source code for a sample user exits that you can use as a basis for your own user exits.

Sample initialization files

These are text files from which the MQSeries link for R/3 server start commands obtain any startup parameters that you do not explicitly specify on the command line.

Sample MQSC files

These are MQSeries command files that create a sample set of MQSeries objects for MQSeries link for R/3.

Sample destination configuration file

The file `smqDestConf` holds the information required to map R/3 logical systems to MQSeries destinations. This file can be edited with a text editor.

SAP initialization files

A sample SAP `saprfc.ini` file is provided to demonstrate the load balancing options required for the inbound server.

Chapter 2. Planning for MQSeries link for R/3

This chapter tells you about the hardware and software you need to run MQSeries link for R/3.

Hardware requirements

To install and run this product, you need a minimum of 5MB of available hard disk space. There are no additional hardware requirements except for those listed for MQSeries and R/3 on the platform you are using. Please consult the installation and planning sections in the appropriate books for these products.

Software requirements

To run MQSeries link for R/3, you must have installed the following:

- At least one R/3 system, Version 3.0E or later
- At least one instance of MQSeries Version 5 with Service level 02 installed for the chosen platform
- The relevant MQSeries product for the chosen platform

Compaq Tru64 UNIX

On Compaq Tru64 UNIX, you need:

- MQSeries for Compaq Tru64 UNIX Server Version 5.1

OS/390

On OS/390, you also need:

- OS/390 Version 2.8, with at least one R/3 system, Version 4.6C or later
- The **librfc** file, supplied with the RFC Software Development Kit that is provided on the SAP presentation CD
- MQSeries Server for OS/390 Version 5.2 with the PTF applied to fix APAR PQ44683.

Windows NT

On Windows NT, you also need:

- The **librfc32.dll** file that is supplied with the SAPGUI component of R/3

Planning

Restrictions

The amount of R/3 IDoc data that can be sent in a single message is limited by the maximum MQSeries message size. With MQSeries Version 5 the maximum message size is 100 MB and with earlier versions of MQSeries the maximum size of MQSeries messages is 4MB. Because the MQSeries link for R/3 header fields are part of the space that is assigned to MQSeries messages, the actual space available for R/3 transaction data is less than the full 100MB (or 4MB). R/3 transaction data can consist of one IDoc or batched IDocs.

SAP AG recommends a maximum of 2MB for each IDoc.

Supported platforms

MQSeries is available for the following minimum release levels of these platforms:

- AIX 4.1.4
- Compaq Tru64 UNIX Version 4.0F and 5.0
- HP-UX 10.10 and 11
- Sun Solaris 2.5.1
- Windows NT 3.5.1 with Service pack 5 installed
- OS/390 Version 2.8
- OS/400 Version 4.2

MQSeries platforms

Using MQSeries link for R/3, you can connect your R/3 system to other platforms that run MQSeries. Currently, you can run MQSeries on the following platforms:

AIX	SINIX and DC/OSx
Compaq Tru64 UNIX	SunOS
Digital VMS VAX	Sun Solaris
Digital open VMS AXP	Tandem NonStop Kernel
HP-UX	UnixWare
NCR UNIX SVR4 MP-RAS	VSE/ESA
OS/2	Windows 3.1
OS/390	Windows 95
OS/400	Windows NT
SCO UNIX	Windows 2000

Chapter 3. Migrating from version 1.0 to version 1.2

This chapter tells you how to migrate to Version 1.2 of MQSeries link for R/3 if you already have Version 1.0 installed. If you do not have a previous version installed go to “Chapter 4. Installing MQSeries link for R/3” on page 15 for the full installation procedures.

Note: No special migration procedures are required to move from Version 1.1 to Version 1.2.

Migration procedure

To migrate to version 1.2 of the MQSeries Link for R/3, you need to copy the contents of the Version 1.0 MQS Options panel (part of the RFC destination configuration) to the Version 1.2 **smqDestConf** file. You complete this task by editing the sample **smqDestConf** file. See “Task 5. Mapping R/3 Logical Systems to MQSeries destinations” on page 35 for details.

If you want to use load balancing you also need to edit the **saprfc** file. See “Task 4b. Mapping inbound server to SAP logon group” on page 35 for details.

Chapter 4. Installing MQSeries link for R/3

This section tells you how to install MQSeries link for R/3 on all the supported platforms. It is assumed that MQSeries has already been installed on your platform.

If you already have MQSeries version 1.0 installed on your system use, the procedures described in “Chapter 3. Migrating from version 1.0 to version 1.2” on page 13 to upgrade to the new version.

READ.ME file

Before starting to install MQSeries link for R/3, review any READ.ME file that may be included on the distribution media. The READ.ME file contains any product and documentation updates that have become available after this book was published.

Installing MQSeries link for R/3 on AIX Version 4.1

It is assumed that MQSeries has already been installed on your platform.

1. Go to SMIT with root authority. From shell, enter `smit`
2. Select the device appropriate for your installation using the following sequence of windows:

```
Software Installation & Maintenance
  Install and Update Software
    Install/Update Selectable Software (Custom Install)
      Install Software Products at Latest Available Level
        Install New Software Products at Latest Level
```

Select the CD-ROM drive as the input device:

```
/dev/cd0
```

3. Press `Do` to display parameters for Install Latest Level.
4. Enter `all_licenced` into Software to install, or select the List button to display the multi-select window.
Set COMMIT Software Updates to NO.
Set SAVE replaced files to YES.
5. Press `Do` to install.

For further information on using `installp` to install software packages, see the AIX documentation.

Installation

Note: If you mount the CD-ROM on /cdrom using smit, the product manual is then available on the CD-ROM in the directory /cdrom/smq/docs.

Installing MQSeries link for R/3 on Compaq Tru64 UNIX Version 4.0F or 5.0

1. Login to your system as root (or su -).
2. Insert the MQSeries link for R/3 CD-ROM in the drive.
3. Create a directory as the mount point for the CD-ROM using the command:

```
mkdir /cdrom
```

4. Mount the CD-ROM on /cdrom. An example command is:

```
mount -r -t cdfs -o noversion,rrip /dev/rz4c /cdrom
```

where the CD-ROM device is located on the c partition of rz4. After mounting the CD-ROM, you can change to the /cdrom directory and view the directories on the CD-ROM.

5. Install MQSeries link for R/3 using the command:

```
setld -l /cdrom
```

6. When installation is complete, unmount the CD-ROM using the command:

```
umount /cdrom
```

You do not have to be a root user to use MQSeries link for R/3 from a Compaq Tru64 UNIX system; an mqm user is sufficient.

Installing MQSeries link for R/3 on HP-UX Version 10

1. Login to your system as root (or su -).
2. Mount the CD-ROM on /cdrom.
3. Use the HP-UX swinstall program to install the software by typing the following command:

```
swinstall -s /cdrom/smq/smq.img
```

For further information on using swinstall to install software packages, see the HP-UX documentation.

Note: The product manual is available on the CD-ROM in the directory /cdrom/smq/docs.

Installing MQSeries link for R/3 on Sun Solaris Version 2.5

1. Login to your system as root (or su -).
2. Check to see if the Volume Manager is running on your system by typing the following command:

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

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

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

substituting `cntndnsn` with the name of your CD-ROM device.

3. Use the Solaris `pkgadd` program to install the software by carrying out the following procedure:
 - a. Type `pkgadd -d /cdrom/smq`

Note: If the CD was running already, substitute the directory structure `/cdrom/smq/smq.img`

- b. Follow the screen prompts.

For further information on using `pkgadd` to install software packages, see the Solaris documentation.

Note: The product manual is available on the CD-ROM in the directory `/cdrom/smq/docs`.

Installing MQSeries link for R/3 on OS/390 Version 5.2

To install MQSeries link for R/3 on OS/390, refer to the Program Directory, GI10-2537.

Note: To provide the appropriate access to the files and programs, use the **chmod**, **chgrp** and **chown** commands in UNIX System Services.

Installing MQSeries link for R/3 on OS/400 Version 4

To install the MQSeries Link for R/3 for AS/400:

1. Insert the product CD in the CD-ROM drive of your AS/400.
2. Issue the following command:

```
RSTLICPGM LICPGM(5733A22) DEV(OPT01)
```

substituting `OPT01` with the actual name of your CD-ROM drive.

Installation

To verify that the product was installed correctly:

1. Issue the Display Software Resources (DPSFWRSC) command to display a list of the software products installed on your system.
2. Page down until you see 5733A22. There should be two entries for 5733A22. If the product is not installed properly you will see ERROR.
3. In case of error, check the job log for any errors that occurred during the install and take any necessary actions before repeating the installation procedures.

After installation the manual will be found in a directory called **books** on a shelf called **shelves**.

Installing MQSeries link for R/3 on Windows NT

The installation procedures on the Windows NT platform, are dependent on the version of Windows NT that is in use. Follow the procedures that relate to your environment.

Installing MQSeries link for R/3 on Windows NT Version 3.5

1. Login to your systems as administrator user.
2. Insert the CD-ROM in the CD-ROM drive, assumed to be e:\.
3. Run SETUP.EXE. One method to do this is as follows:
 - a. From the Program Manager, select the File menu and click on Run.
 - b. A window titled Run is displayed. In the Command Line field, type:
e:\setup.exe

Press ENTER or click on OK.

4. Select the national language that you want to install by clicking on the corresponding button.
5. Follow the screen prompts.

Installing MQSeries link for R/3 on Windows NT Version 4.0

1. Login to your systems as administrator user.
2. Insert the CD-ROM in the CD-ROM drive, assumed to be e:\.
3. SEUP.EXE should start automatically, displaying the national language panel.

If it does not start automatically, run SETUP.EXE manually as follows:

 - a. On the Windows desktop, click on Start and select Run from the menu.
 - b. A window titled Run is displayed. In the Open field, type:
e:\setup.exe

Press ENTER or click on OK.

4. Select the national language that you want to install by clicking on the corresponding button.
5. Follow the screen prompts.

Note: The product manual is available on the CD-ROM in the directory \smq\docs.

Directories after installation

For each platform, the directory structures that will exist after installation are as listed below.

Notes:

1. In the directory structures *lang* is the directory name for the national language version that is installed.
2. The /samp directory contains sample initialization files for the outbound and inbound servers and also an example **smqDestConf** file. We recommend that you use these files as templates during the configuration procedures.

AIX

```

/usr/lpp/smq/bin
/usr/lpp/smq/include
/usr/lpp/smq/lib
/usr/lpp/smq/samp
/usr/lib/nls/msg/lang/smq.cat

```

Compaq Tru64 UNIX

```

/opt/smq/bin
/opt/smq/include
/opt/smq/lib
/opt/smq/samp
/usr/lib/nls/msg/lang/smq.cat

```

HP-UX

```

/opt/smq/bin
/opt/smq/include
/opt/smq/lib
/opt/smq/samp
/usr/lib/nls/msg/lang/smq.cat

```

Sun

```

/opt/smq/bin
/opt/smq/include
/opt/smq/lib
/opt/smq/samp
/usr/lib/locale/lang/LC_MESSAGES/smq.cat

```

Installation

OS/390

```
/usr/lpp/smq/bin  
/usr/lpp/smq/include  
/usr/lpp/smq/samp  
/usr/lib/nls/msg/lang/smq.cat
```

where these files reside on a hierarchical file system (HFS).

Samples are also installed as members of the partitioned data sets: SASQJCL and SASQASM. Refer to “OS/390 samples” on page 103 for a description of the members.

OS/400

```
/QIBM/ProdData/smq  
/QIBM/ProdData/smq/MRI<nnnn>/smq.cat  
/QIBM/ProdData/smq/samp  
Product Library:QMQLINK
```

where *nnnn* is the feature code for your specific language.

Windows NT

```
\smq\bin  
\smq\include  
\smq\dll  
\smq\samp  
\smq\bin\smqcatnt.dll
```

Uninstalling MQSeries link for R/3 from AIX

Use SMIT to remove all the MQSeries link for R/3 files that were installed.

Uninstalling MQSeries link for R/3 from Compaq Tru64 UNIX

To remove MQSeries link for R/3 from a Compaq Tru64 UNIX system, you must login as a root user and then issue the command:

```
setld -d SMQ_BASE
```

Uninstalling MQSeries link for R/3 from HP-UX

To remove an MQSeries client from your HP-UX system, use the `swremove` command, or use SAM. You can then delete the `/smq` directory tree.

Uninstalling MQSeries link for R/3 from Sun Solaris

To remove an MQSeries client from your Sun Solaris system, use the `pkgm` program.

Uninstalling MQSeries link for R/3 from OS/390

To remove the MQSeries link for R/3 product from your OS/390 system, refer to the Program Directory, GI10-2537.

Uninstalling MQSeries link for R/3 from OS/400

To remove the product from OS/400 issue the Delete Licensed Program command, DLTLICPGM as follows:

```
DLTLICPGM LICPGM(5733A22)
```

Uninstalling MQSeries link for R/3 from Windows NT

This section describes how to remove MQSeries link for R/3 from a workstation that is running Windows NT. The procedure is dependent on which version of NT you are running.

Before attempting to remove the product, logon to the system as an administrator.

If you want to reinstall the MQSeries link for R/3, you must restart your computer after the removal procedures.

Removing MQSeries link for R/3 from Windows NT Version 3.5

To remove MQSeries link for R/3:

1. Double-click on the "IBM MQSeries link for R/3" program folder group from the program manager.
2. A window titled "IBM MQSeries link for R/3" is displayed.
3. Double-click on the icon "Uninstall IBM MQSeries link for R/3".
4. Follow the on-screen instructions.

Removing MQSeries link for R/3 from Windows NT Version 4.0

To remove MQSeries link for R/3:

1. On the Windows desktop, click on Start.
2. Select Settings and open the Control Panel.
3. Double-click on Add/Remove Programs.
4. Select "IBM MQSeries link for R/3" from the list of installed software.
5. Click on Add/Remove.
6. Follow the on-screen instructions.

Uninstalling

Chapter 5. Configuring MQSeries link for R/3

Before you can use MQSeries link for R/3, you must configure it to ensure that messages end up at the correct destinations. Before you can do any of this, you must understand the reasons for the configuration. These are explained in:

- “Understanding which MQSeries objects you need”
- “Advanced queue manager configuration” on page 28
- “Understanding RFC destinations” on page 29

If you are running MQSeries link for R/3 on OS/390, the section “Configuring MQSeries link for R/3 for OS/390” on page 30 gives you an overview of accessing the UNIX shell on OS/390.

The tasks themselves are described:

- “Task 1. Defining TCP/IP Ports for use with the operating system” on page 31
- “Task 2. Creating the MQSeries objects” on page 31
- “Task 3. Defining the RFC destinations on R/3” on page 32
- “Task 4. Setup for inbound server load balancing” on page 34
 - “Task 4a. Creating a SAP logon group” on page 34
 - “Task 4b. Mapping inbound server to SAP logon group” on page 35
- “Task 5. Mapping R/3 Logical Systems to MQSeries destinations” on page 35
- “Task 6. Specifying server configuration in the initialization (ini) files” on page 39
- “Task 7. Starting the servers” on page 40

When you have configured your system, you can then test the configuration, as described in “Testing the basic configuration” on page 41.

Understanding which MQSeries objects you need

Before you can start either the outbound or the inbound server, you must define the MQSeries objects that MQSeries link for R/3 is going to use. You do this using the MQSeries commands (MQSC commands) supplied with MQSeries.

For more information about MQSeries:

- The *MQSeries System Administration Guide* or *MQSeries System Management Guide* for your platform provides information about setting up queue managers and running MQSC commands to create MQSeries objects.

Configuring the link

- The *MQSeries MQSC Command Reference, SC33-1369* provides information about the individual MQSC commands.
- The *MQSeries® Intercommunication* book, SC33-1872, provides information about setting up channels between queue managers.

Defining a queue manager

You need to define some MQSeries queues that the MQSeries link for R/3 servers use at run time. However, first you must create a queue manager, if you do not already have one. For outbound and inbound servers, you need to specify the name of the queue manager in the Initialization file. The server automatically connects to the queue manager you specify and opens the queues it needs for input or output, as required.

Recommendation

Use the same queue manager for the queues used by both outbound and inbound servers. This is not a requirement, but it does simplify the configuration.

Data conversion exit library

The data conversion exit library MQHSAP is shipped with MQSeries link for R/3. On the UNIX platforms a link is created for this library in `/usr/lib`. To avoid any problems when converting messages, make sure this is a directory in your `LIBPATH` when you run the inbound server.

If you are using an MQSeries Version 5 server, the path to the Data conversion exit must be specified in the MQSeries `QM.INI` file. For details of entries in this file see *MQSeries Intercommunication, SC33-1872*.

Defining MQSeries objects for an outbound server

For an outbound server, you must define two (MQSeries) queues:

Outbound message queue

The queue that receives messages from R/3 applications. You must specify the name of this queue when you start the server using `smqso` command.

You can define this queue as a local queue or a remote queue (see “Choosing an outbound queue type” on page 25 for more information).

Outbound transaction ID queue

This is a local queue that MQSeries link for R/3 uses to maintain a record of the transaction IDs within a unit of work. This queue is used by MQSeries link for R/3 if it becomes necessary to back out a unit of work.

Choosing an outbound queue type

You can define your outbound queue as a local or remote queue.

If you are sending MQSeries messages between R/3 systems, you can use a remote queue definition that identifies the inbound message queue on the workstation that is going to receive the messages. This means that you do not need an application program to handle these messages. However, you do not have much flexibility with respect to the destination, because all messages go to the same place.

If you define the outbound queue as a remote queue, you must specify the following objects on the queue manager that is sending the MQSeries messages:

- The transmission queue for this remote queue. This is a local queue with its *USAGE* attribute specified as 'XMIT'.
- A sender channel definition. Typically, your MQSeries messages are sent to another queue manager. To do this, you must define a sender channel on the local queue manager. This definition must be consistent both with the channel definition at the receiving end and with the definition of the transmission queue for the channel.

It is also possible to use remote queues in conjunction with queue manager aliasing (see *MQSeries Intercommunication*, SC33-1872, for more information on aliasing).

Figure 7 on page 27 shows an MQSeries configuration that uses a remote queue as the outbound queue.

Defining MQSeries objects for an inbound server

For an inbound server, you must define the following MQSeries queues for a queue manager on the inbound server system:

Inbound message queue

The queue that receives messages from other applications. This queue must be unique for each server instance.

Inbound transaction ID queue

Typically, this is a local queue that MQSeries link for R/3 uses to maintain a record of the transaction IDs within a unit of work. This queue is used by MQSeries link for R/3 if it is necessary to back out a unit of work. This queue must be unique for each server instance.

Bad message queue

This queue is used to store inbound messages that the inbound server has attempted to process, but failed. For more information, see "Handling unrecognizable messages" on page 86. This queue can be shared by many server instances.

Configuring the link

Note: In the ini file, you can specify whether the server stops if a bad message is received.

You also need to define a receiver channel or equivalent. Typically, your inbound MQSeries messages come from another queue manager. To be able to receive these messages, you must define a receiver channel on the local queue manager. This definition must match the channel definition at the sending end. For more information about channels, see *MQSeries Intercommunication*.

Using the Sample MQSC command file

MQSeries link for R/3 provides a sample MQSC command file, **smqscdef.tst**. For more details of the sample see “MQSC command file (smqscdef.tst)” on page 102. When you run this file, using the *runmqsc* command, it creates a set of queues corresponding to entries in the sample ini files supplied.

Note: This file does not support the example shown in Figure 7 on page 27.

On OS/400 the sample MQSeries commands are in the file **QMQSC** in library QMQLINK in the member SMQSCDEF. To run the commands in this file use the MQSeries command *STRMQMMQSC* as follows:

```
STRMQMMQSC SRCMBR(SMQSCDEF) SRCFILE(QMQLINK/QMQSC)
```

Example of an MQSeries configuration

Figure 7 on page 27 shows one R/3 application sending information (in the form of IDocs) to an R/3 application on another system. In this example, the R/3 systems communicate using MQSeries link for R/3 and MQSeries messages. This example shows how both the inbound and outbound servers can work in relation to MQSeries.

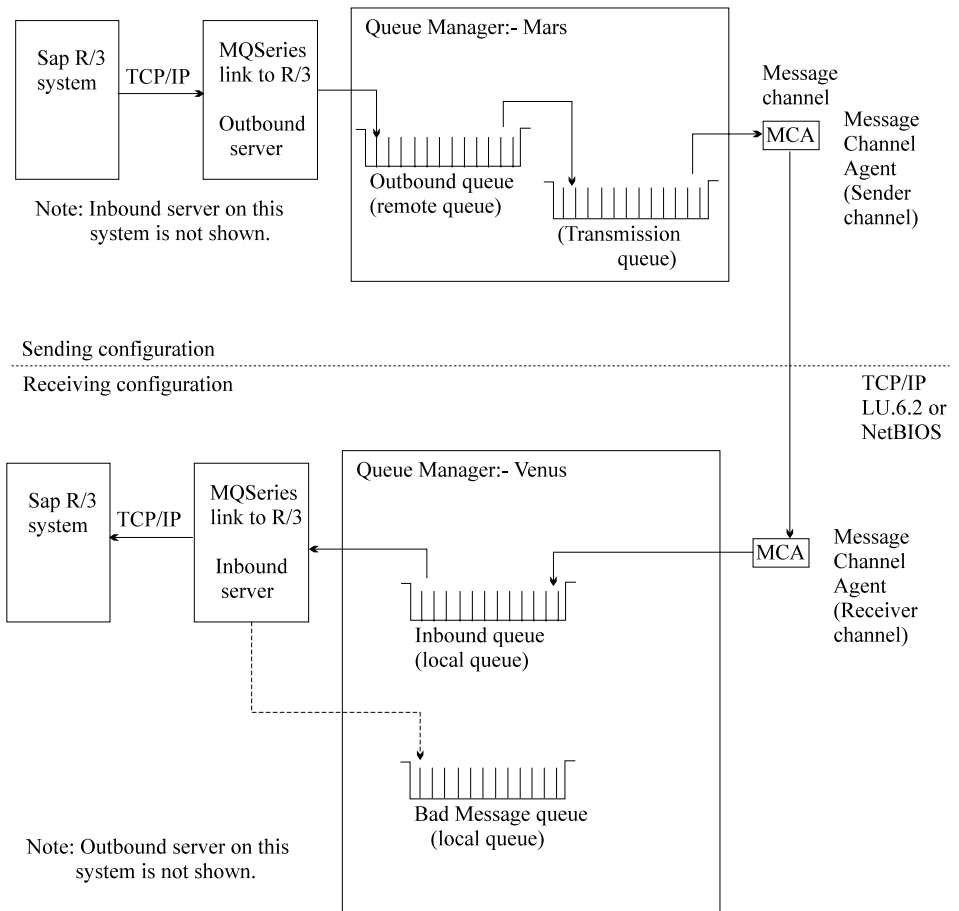


Figure 7. MQSeries link for R/3 and MQSeries. In this case, the MQSeries link for R/3 outbound queue is defined as a remote queue that identifies a local queue on another queue manager.

On the outbound server

In Figure 7, the sending R/3 application sends IDocs to the MQSeries link for R/3 outbound server that you have defined for it. The server creates an MQSeries message containing the IDocs as the application data part of the message. It also adds a link header to the message; this header contains information that the inbound server uses at the other end. The outbound server puts the MQSeries message it has created onto its outbound queue.

The outbound queue has been defined as a remote queue object (more accurately, a local definition of a remote queue). A remote queue object identifies a local queue on another system, that is, the inbound queue for the inbound server. You don't have to do this, but it is convenient because the message ends up on the other system, where it can be accessed by the

Configuring the link

inbound server on that system. Remember that MQSeries applications can put messages on local or remote queues, but can only retrieve messages (using *MQGET*) from a local queue.

On the inbound server

The inbound server decomposes the messages into its constituent IDocs and passes these on to the destination R/3 application, as specified in the MQSeries link for R/3 header in the MQSeries message itself, or in the configuration information defined for the inbound server. Any messages the server cannot process are sent to the *bad message queue*. By making the appropriate statement in the inbound server ini file, you can specify whether the server stops when it gets a bad message.

Advanced queue manager configuration

In MQSeries, you can define more than one queue manager for each workstation on most platforms. MQSeries link for R/3 allows you to specify different queue managers for each instance of a server. For outbound servers, you can specify different queue managers for different queues, as explained below.

Outbound servers

For an outbound server, there are two queues for each instance of a server and you can, if necessary, put these on different queue managers. The queues for an outbound server are:

- Outbound queue
- Outbound transaction ID queue

You specify the name of the queue manager that owns the outbound queue in the *Queue manager name* field in the destination configuration file **smqDestConf**. You specify the name of the queue manager that owns the transaction ID queue, either as a parameter on the **smqso** (start outbound server) command or as a parameter in the outbound ini file. If you do not specify a queue manager explicitly, MQSeries link for R/3 uses the default queue manager.

Note: You cannot use the same queue for both functions. You must define them as distinct queues, or MQSeries link for R/3 cannot function properly.

Inbound servers

For an inbound server, all the queues for MQSeries link for R/3 are owned by the same queue manager. These queues are:

- Inbound queue
- Inbound transaction ID queue
- Bad message queue

Note: You cannot use the same queue for different functions. You must define them as three separate queues, or MQSeries link for R/3 cannot function properly.

Understanding RFC destinations

To enable an R/3 system to send information over MQSeries link for R/3, you must specify the RFC destination using an R/3 transaction in the ALE layer.

You use MQSeries link for R/3 to transfer data, initially in the form of IDocs, from an R/3 system to another system, which can be:

- Another R/3 system
- An R/2 system
- A non-SAP system

On the outbound server, the IDoc data is put into an MQSeries message that is then put on the outbound server queue. You must identify this queue in the destination configuration file **smqDestConf** using the sample **smqDestConf** file that is provided with MQSeries link for R/3.

If the final destination is another R/3 system, the IDocs contained in the message must be sent to the correct R/3 destination. To ensure this, you must specify the destination information either in the appropriate R/3 panel or in the **ini** file of the inbound server that is connected to the destination R/3 system.

If the destination is a non-SAP system, you do not have to specify the SAP destination information.

Specifying a remote R/3 client

The ALE layer on the sending R/3 system uses the parameters specified in the RFC Destination panels to connect to the outbound server. It uses the same principle whether the outbound server is local to the sending system or remote (across a network). Typically, there are multiple gateway servers across the network, all serving the sending R/3 system.

On one of these servers the *hostname* must be associated with a *program Id*, where *hostname* is the physical address of the machine on which the outbound server runs, and *program Id* is a unique parameter that links the sending R/3 system to a particular instance of the outbound server. Therefore, you have to specify this *program Id* in both the RFC destination panels and on the start server command.

Configuring MQSeries link for R/3 for OS/390

On OS/390, MQSeries link for R/3 runs in the UNIX shell. The section “Accessing the OS/390 shell” gives you an overview of how you can access the OS/390 shell. For more detailed information, refer to the *OS/390 UNIX System Services User’s Guide*, SC28–1891.

The section “Starting the MQSeries queue manager” tells you about operating MQSeries in an OS/390 environment.

Accessing the OS/390 shell

You can access OS/390 UNIX services in any of the following ways:

- Using JCL to run the MVS BPXBATCH utility. BPXBATCH is an MVS utility that you can use to run shell commands or shell scripts and to run executable files through the MVS batch environment.
- Using the OMVS command facility. To invoke the OS/390 shell, log on to TSO/E and then enter the TSO/E OMVS command.
- Using rlogin. When the **inetd** daemon is set up and active, you can rlogin to a shell from a workstation that has rlogin client support and is connected via TCP/IP or Communications Server to the MVS system. To login, use the rlogin command syntax supported at your site.

Note: When the shell is first invoked you are placed in the home directory: `/u/system_user_name`, where `system_user_name` is your user root directory.

Starting the MQSeries queue manager

You control the MQSeries operational environment with MQSeries commands. For details of the syntax of these commands, refer to the *MQSeries MQSC Command Reference*. You can issue these commands from an OS/390 console or its equivalent. In this environment each MQSeries command must be prefixed with a command prefix string. The examples in this section use the string `+CSQ1`, where `+` is the command prefix and `CSQ1` is the name of the queue manager.

You access the OS/390 console from the TSO ISPF menu by selecting the SDSF facility. Use the following command format to start your queue manager:

```
+CSQ1 START QMGR
```

Direct responses to commands are sent to the MVS Console log, which you access from the TSO ISPF menu SDSF facility LOG option. A correctly started queue manager will return a response similar to that below:

```
CSQ1MSTR - STARTED - TIME=16.09.38
```

Configuration tasks

The following sections describe the tasks for configuring your environment to run MQSeries link for R/3.

Task 1. Defining TCP/IP Ports for use with the operating system

To allow the outbound server to communicate with the R/3 system, the following TCP/IP ports must be defined in the local system:

sapdpnn

32nn/tcp

sapgwnn

33nn/tcp

where *nn* is the system instance number of your source R/3 system.

These TCP/IP port definitions must be placed into the following files according to your platform:

Platform	path/filename
AIX	/etc/services
Compaq Tru64 UNIX	/etc/services
HP-UX	/etc/services
OS/390	/etc/services
Sun Solaris	/etc/services
Windows NT	...\WINNT35\system32\drivers\etc\services

Note: On OS/400, use the command `wrksrvtble` to see a list of TCP/IP ports and add new ports, if necessary.

Task 2. Creating the MQSeries objects

The following procedure assumes that you are using *one* queue manager.

To create the MQSeries environment for MQSeries link for R/3 on:

AIX, Compaq Tru64 UNIX, HP-UX, Sun Solaris, and Windows NT

1. If you have not already done so, create a queue manager for the queues (and channels) that MQSeries link for R/3 requires.
Use the `crtmqm` command to do this.
2. If the queue manager is not running, start it with the `strmqm` command.

Configuration tasks

3. Create the queues and channels from an MQSC command file by running the following command:

```
runmqsc qmname<smqscdef.tst
```

where qmname is the name of the queue manager and smqscdef.tst is the file that contains the required MQSC DEFINE commands.

4. If necessary, start the channel for sending messages to other queue managers.

OS/390

Use the **CSQUTIL** utility program provided with MQSeries for OS/390 to configure MQSeries link for R/3. The following sample JCL shows you how to use **CSQUTIL** to define the required MQSeries objects:

```
//COMMAND EXEC PGM=CSQUTIL,PARM='CSQ1'  
//STEPLIB DD DISP=SHR,DSN=th1qual.SCSQANLE  
// DD DISP=SHR,DSN=th1qual.SCSQAUTH  
//CSQUCMD DD DSN=SASQJCL(ASQCMS),DISP=SHR  
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *  
COMMAND  
/*
```

where:

- CSQ1 is the name of your queue manager
- th1qual is the high level qualifier of the MVS data set for MQSeries
- CSQUCMD is the reference name for the data set containing the MQSeries commands
- SASQJCL is the supplied library containing sample JCL for MQSeries link for R/3 on OS/390

For more information, refer to the *MQSeries for OS/390® System Administration Guide*.

OS/400

Use the following command to run the MQSeries command file:

```
STRMQMMQSC SRCMBR(SMQSCDEF) SRCFILE(QMQLINK/QMQSC)
```

Task 3. Defining the RFC destinations on R/3

Note: You must configure the Application Link Enabling (ALE) of your SAP system for sending and receiving IDoc data. Refer to your SAP documentation for information about configuring the ALE. In particular, the outbound server requires a unique program id to register to SAP. This part of the configuration is described below.

To set up an outbound destination object from the ALE layer, use the SAP RFC configuration transaction (sm59).

1. In any R/3 panel, specify administration function sm59, by typing /nsm59 in the command field to display the 'Display and Maintain RFC Destinations' panel.
2. Click on the 'Create' button in this panel to display the 'RFC destination panel'.
3. Complete the fields in the 'RFC destination panel' as follows:

RFC destination

specify a destination name. This can be any name you like, but it must *not* contain any embedded spaces.

Connection type (required)

specify T (for TCP/IP).

Description

Type some descriptive text for this connection

Save the settings.

4. Click the 'Registration' button. This changes the panel to allow you to enter the program id for your outbound server connection.
5. In the **program ID** field, type in a valid program ID. This is the name by which this particular outbound server connection will be known. You can call it what you like, but it must match the value you specify when you start the outbound server. The server gets the value from the command line or the outbound **ini** file. (The command line takes precedence over the ini file).
6. Choose the 'Gateway options' from the 'Destination' menu (on the menu bar) to display a pop-up panel in which you specify the gateway host and service.
7. In this panel, specify values for the **Gateway host** and the **Gateway service** fields. These values should already be defined for your system. If you do not know what to put in here, talk to your SAP administrator.
8. Occasionally the R/3 system may fail to connect to the MQSeries link for R/3 outbound server, causing IDocs to be delayed. These IDocs are shown in R/3 transaction sm58 with the message 'Gateway or Target system not active'. To force R/3 to automatically retry these IDocs, pull down the 'Destination' menu and select 'TRFC options' to display a panel in which you can specify the IDoc retry parameters. Fill in values for the 'number of retries' and the 'retry interval'. Some experimentation may be required to find values that provide optimum performance on your system.

It is also recommended that you reduce the single connection attempts to the R/3 system by increasing the number of IDocs that can be placed in one MQSeries message. For example, batch 10 IDocs together.

Task 4. Setup for inbound server load balancing

Note: This task is optional. You need to perform this only if you are using multiple application servers.

If you want to use load balancing on the inbound server you need to carry out the following configuration steps:

1. Create a SAP logon group
2. Map the inbound server to the SAP logon group

The procedures for these two tasks are described in “Task 4a. Creating a SAP logon group” and “Task 4b. Mapping inbound server to SAP logon group” on page 35.

Task 4a. Creating a SAP logon group

To create logon groups within SAP, use the CCMS:Maintain Logon Groups (sm1g) transaction as follows:

1. Within an R/3 window, specify /nsmlg in the command field to create a new session for logon group maintenance.
2. Click on 'Create entry' in the application toolbar.
3. Complete the entry fields in the logon group panel as follows:
 - a. in the **Group allocation** dialog box specify:

Logon group

You can use any name you like

Instance

The servername_systemname_systemnumber

- b. If you want to set the 'Front-End instance Connection' and 'Load Limit' click the **More information** button. (Otherwise go to step 3c on page 35) Complete the fields as follows:

Response time

This is used to specify an upper bound logon constraint. A default value for all groups within an instance is set automatically, but once a response time has been defined for one group in an instance you must define a response time for all groups that use the same instance.

Users The maximum number of users that may be logged on to an instance. A default value for all groups within an instance is set automatically.

The values of these two fields are not hard constraints. It is possible to log on to another instance if the current values are higher than the configured values. They only influence the calculation of the current logon server of the logon groups.

- c. Click the **Copy** button to save the logon group information.
4. Repeat this configuration with as many application server entries for that logon group as needed.

Task 4b. Mapping inbound server to SAP logon group

The **saprfc.ini** file is required by SAP to capitalize on the load balancing features available from SAP R/3 version 3.0 and above. The file can be placed either in the same directory as the inbound server or it can be defined with the *RFC_INI* environment variable. “saprfc.ini file information” on page 80 describes the **saprfc.ini** file parameters.

Environment variable examples:

Platform	Settings
AIX	export RFC_INI=<Value> ;for ksh shell setenv RFC_INI<Value> ;for csh shell
AS/400	ADDENVVAR ENVVAR(RFC_INI) VALUE('/QSYS.LIB/R340ARFC.LIB/INI.FILE/SAPRFC.MBR')
HP-UX	Shell Setting the Variable .sh RFC_INI=<value> export RFC_INI ksh export RFC_INI=<value> csh setenv RFC_INI <value>
Sun Solaris	% setenv RFC_INI path_to_saprfcini where <i>path_to_saprfcini</i> is the path to the directory where the saprfc.ini file is stored.
Windows	set RFC_INI=c:\mqm\link\src\saprfc.ini

Task 5. Mapping R/3 Logical Systems to MQSeries destinations

When the RFC destination has been configured, the outbound server is ready to start receiving IDocs.

The outbound server uses information in the control record of each IDoc to determine which MQSeries Queue and Queue Manager to forward the message to.

Note: If batching is in use and a batch of IDocs is received, the control record of the first IDoc in the batch is used. This is why all IDocs in a batch should be of the same IDoc type.

The mapping of R/3 logical destinations to MQSeries destinations is held in a configuration file, **smqDestConf**, that is read by the outbound server during start up. A number of different mappings can be defined to enable different

Configuration tasks

IDoc types to be sent to different logical systems and queues. Each destination map must contain the IDoc type, the destination R/3 logical system name and the outbound server queue that should be used. A number of other parameters can also be set to control the MQSeries and R/3 resources that should be used by the message type. In addition one mapping should be designated as the default to be used if an IDoc/Receiving destination combination is received for which no specific mapping is defined. To indicate the default destination mapping, include the *default* parameter set to 'yes' in the definition. A sample file is provided in the /samp subdirectory of your installation, use this file as a template to edit with your own definitions. The edited file should be saved as **smqDestConf** in the same directory as the outbound server executable (**smqso.exe**). Figure 17 on page 98 shows details of the **smqDestConf** file.

Note: The 'MQS Options panel' in SAP is no longer used to gather this information.

Configuration parameters in the smqDestConf file

The following parameters can be set in each of the destination mapping definitions:

General parameters

Receivingpartner (required)

This is the value of the R/3 logical system as entered in the **RCVPRN** field of the EDI_DC structure of the IDoc control block. This field is 10 bytes long.

edi_mestype (required)

This is the edi_mestype as entered in the **MESTYP** field of the EDI_DC structure of the control block of the IDoc. This field is 6 bytes long for SAP R/3 Version 3 and 30 bytes long for SAP R/3 Version 4.

The edi_mestype represents the logical message type, that can map to many IDoc types.

You can put an asterisk in this field indicating that these settings apply to all edi_mestypes for this receiving destination.

Note: If more than one MQSeries destination is defined, the outbound server uses the first destination that matches the IDoc parameters. You can set an MQSeries destination for all edi_mestypes by specifying edi_mestype=* in the configuration file, but this destination should come after your settings that match specific IDoc parameters.

outboundqueue (required)

The name of the outbound message queue used by the outbound

server. After generating the MQSeries message that includes the IDoc information, the outbound server puts the complete message on the outbound server queue.

outboundqueuemanager

The name of the queue manager that is associated with the outbound server and that owns the outbound server queue. The queue manager name can exist as an alias to the queue manager that owns the outbound queue. (See *MQSeries Intercommunication*, SC33-1872, for more information on queue manager aliasing.)

If you do not specify a name here, the queue manager name is taken from the *-m* option on the *smqso* (start outbound server) command. If you omit the *-m* option, the name is taken from the **ini** file. If the queue manager name is not specified in any of these places, the default queue manager name is used.

exitname

The name and path of the user exit executable. To call the exit, you must also set *calluserexit* to 'yes'.

Alternatively, if you type an asterisk (*) here and set *calluserexit* to 'yes', the user exit name in the **ini** file is used.

exitbuffer

A flag containing up to 32 bytes of data that is passed to the user exit at run time. How you use this flag is up to you. For example, you can use this data to provide input data for a switch in the user exit where *test* tells the exit that this is a test run.

calluserexit

Set this to 'yes' if you want to call the user exit on the outbound server. Otherwise set it to 'no'.

Remote R/3 connection parameters

If you are sending outbound messages to a remote R/3 system, you can include the remote destination information with the following parameters.

Note: This information can also be defined in the inbound **ini** file of the remote destination.

client The client id of the destination R/3 system.

language

The language identifier of the destination R/3 system.

userid

The R/3 user ID of the destination R/3 system.

password

The password of the destination R/3 system. You are recommended not to use this field, see "Security precautions" on page 93.

Configuration tasks

hostname

The application server of the destination R/3 system.

systemnumber

The system instance number of the destination R/3 system.

Designating the default destination mapping

Include the following parameter set to 'yes' in the mapping definition that should be used if a Receiving destination/IDoc combination is received for which no specific mapping is defined.

default

The MQSeries destination settings that should be used for any Receiving destination/IDoc type combination that is not defined in the configuration file.

Notes:

1. This parameter must be included in one and only one definition.
2. If default=yes is not found in the file, the link to R/3 fails.

General notes

1. If you do not know any of the R/3 connection parameters, contact your R/3 administrator.
2. Any 'Remote R/3 connection parameters' that you leave blank default to the values set on the inbound server on the remote system.
3. Much of this information is contained in the link header of the message. You should be aware that, if you define a password in the RFC destination panel, the password of the remote R/3 system is stored in this header. See "Security precautions" on page 93.
4. To test that the connection is working correctly, start the outbound server and, in the RFC destination panel, use the Test Communication button.
5. The program ID, gateway host, and gateway service uniquely identify an outbound server.
6. While the outbound server is running you can see the registration of the outbound server destination object using transaction SMGW. This shows the TCP/IP address of the outbound server machine and the connection type.
7. The details from the **smqDestConf** file are only implemented when a complete batch of IDocs has been given to the outbound server. This means that if you make changes to the **smqDestConf** file after IDocs have been created, and these IDocs are being collected as part of a batch, then the changes to the file affect all IDocs in the batch, not just to the IDocs that are created after the changes are made.

Task 6. Specifying server configuration in the initialization (ini) files

You can specify server configuration parameters in the **ini** file associated with that server. Typically, you use the **ini** files to set parameters that you do not change very often. You can specify a subset of these parameters on the command line.

Figure 14 on page 95 and Figure 15 on page 96 show the parameters in the sample ini files for outbound and inbound servers. To find out more about these parameters and what they do, see “Inbound server initialization parameters” on page 75 and “Outbound server initialization parameters” on page 71.

Note: The **ini** file is required for an inbound server, but optional for an outbound server.

Initializing the outbound server

When you start the outbound server, the following parameter values, specified in the RFC destination panels **must match** the equivalent parameter values specified in the **ini** file for the outbound server, or in the command line parameters for the outbound server:

- *ProgramID*
- *GatewayHost*
- *GatewayService*

If these values do not match, the server will not be able to establish a successful connection to the R/3 system.

The outbound server puts the initialization information from the **smqDestConf** parameters described in “Remote R/3 connection parameters” on page 37 into the header of the MQSeries message.

Figure 14 on page 95 shows a sample outbound **ini** file.

Note: The *exitname* parameter is only used if *userexit* is set to '*' and *calluserexit* is set to 'yes' in the **smqDestConf** file.

Initializing the inbound server

The inbound server must connect to an R/3 system and to the queue manager that owns the queues associated with the server. To do this, it uses information that may be obtained from:

- The command line parameters specified with the `smqs i` command.
- The inbound **ini** file.
- The header of the message to be transmitted. This information, which is optional, is defined when you create an RFC destination.

Figure 15 on page 96 shows a sample inbound **ini** file.

Configuration tasks

Task 7. Starting the servers

If you are running MQSeries link for R/3 on OS/390, you must complete the steps in “Creating the MQSeries environment for OS/390”. For all other platforms, move directly to “The start commands”.

Creating the MQSeries environment for OS/390

To create the MQSeries environment for MQSeries link for R/3:

1. Include the location of the `librfc` library in your `LIBPATH`:

```
export LIBPATH=sap_library/lib:$LIBPATH
```

where `sap_library` is the path of the SAP library.

2. Ensure the product message catalog (`smq.cat`) is in the location pointed to by the `NLSPATH`, which should already be set to:

```
/usr/lib/nls/msg/lang/%N.cat
```

where `lang` is the national language version that is installed.

3. Set `SAP_CODEPAGE="0100"`.
4. When you are running from UNIX System Services:

```
export STEPLIB=th1qua1.SCSQANLE:th1qua1.SCSQAUTH
```

where `th1qua1` is the high level qualifier of the MVS data set for MQSeries.

Note: When the MQSeries link for R/3 is invoked using JCL, ensure that the following MQSeries data sets are included in the STEPLIB:

```
//STEPLIB DD DISP=SHR,DSN=th1qua1.SCSQANLE  
// DD DISP=SHR,DSN=th1qua1.SCSQAUTH
```

The start commands

Note: If you want to use load balancing, you must change the current directory to the directory containing the appropriate `saprfc.ini` file before starting the inbound server.

You start both the outbound and inbound servers using the supplied commands. The servers can receive initialization information from three separate sources. If the same information is specified in more than one place, the priority is assigned as follows:

1. Command line parameters specified on the start server commands.
2. The initialization files. (You specify the appropriate ini file name on the start server commands.)

The commands to start the server are:

smqso - Start the outbound server.
smqsi - Start the inbound server.

When you start an MQSeries link for R/3 server, you must specify an **ini** file, the other parameters are optional. The minimum forms of the commands are:

```
smqso -iout.ini  
smqsi -iin.ini
```

where **in.ini** and **out.ini** are the respective initialization files for the inbound and outbound servers.

Notes:

1. On OS/390, you can use the BPXBATCH program to manage the link servers from an MVS batch job. Customise the following members in the supplied SASQJCL library to suit your installation:

ASQRUN

To start the link server

ASQSTOP

To stop the link server

ASQQUERY

To query the status of the link server

2. On OS/400, the parameters for the commands smqsi and smqso must be enclosed in single quotes. For example:

```
smqso '-iout.ini'  
smqsi '-iin.ini'
```

To find out more about these commands and the parameters you can specify on them, see “Chapter 7. Command reference” on page 59.

Starting multiple instances of the outbound server.

If you want to start multiple instances of the outbound server to help the flow of IDoc processing, (see “Running multiple outbound servers to the same program id” on page 5), the instances should all be started from the same directory and they should connect to SAP using the same *programid* name. The servers should all be started with the same configuration information and they should use the same Transaction Queue.

Testing the basic configuration

Note: This section assumes an R/3 to R/3 communication over MQSeries link for R/3.

You can now test the configuration by running through the following steps:

1. Start an outbound server, connecting it to the sending R/3 system.

Configuration testing

2. Start an inbound server, connecting it to the destination (remote) R/3 system.
3. Send some IDocs.
4. Monitor the IDoc traffic.

1. Starting the outbound server

1. Check the critical server initialization parameters:
 - *ProgramID*
 - *GatewayHost*
 - *GatewayService*

These must match the defined parameters in the R/3 RFC destination panel configuration for an outbound server.

2. Make changes, if necessary, to the outbound initialization file,
3. Change your current directory to be the directory where the **smqDestConf** file resides,
4. Start the outbound server on the host machine using the command (this example is for the Windows NT platform):

```
smqso -id:\smq\out.ini
```

Note: On OS/400, the parameters for the `smqso` and `smqsi` commands must be enclosed in single quotes and forward slashes must be used for the path.

2. Starting the inbound server

1. Check that all the parameters in the initialization file point to valid objects in the R/3 and MQSeries configurations.
2. Make changes, if necessary, to the inbound initialization file.
3. Start the inbound server on the host machine. (This example is for the AIX platform.)

```
smqsi -i/usr/lpp/smq/in.ini
```

Note: On OS/400, the parameters for the `smqso` and `smqsi` commands must be enclosed in single quotes and forward slashes must be used for the path.

3. Sending the IDocs

Note: This step assumes that the necessary SAP and ALE configuration has been performed to enable the IDocs to be sent

To do this follow a procedure similar to the one described below for sending MATMAS IDocs (R/3 transaction `bd10`), as described in your SAP documentation:

1. Type in the following data (or its equivalent on your system):

Field	Data
Material	mq_coco
edi_mestype	MATMAS
Receivingpartner	MQTESTLSYS

2. Click on **Enter** to send the data.

4. Monitoring IDoc traffic

You can monitor the IDocs coming in and going out of an R/3 system using R/3 transaction we05 or its equivalent, as described in your SAP documentation:

1. Select the period of interest for IDoc traffic monitoring (the default is for today).
2. Click on **Enter** to see a list of inbound and outbound IDocs.
3. To view the status of any of the IDocs displayed in the colored table, double click on the IDocs of interest. If the IDocs do not appear to be reaching the outbound server you can also use transaction sm58 to view any failures.

Configuration testing

Chapter 6. Writing user exits

This chapter tells you how to write user exits for the MQSeries link for R/3 servers. It also describes the structures of MQSeries messages that contain IDocs and of the MQSeries link for R/3 header. This chapter contains these sections:

- “About user exits”
- “Understanding the external interface” on page 48
- “Sample user exits” on page 53
- “Compiling user exits” on page 53
- “Message formats” on page 55

For more information about MQSeries messages and their structure, refer to the *MQSeries Application Programming Reference, SC33-1673*.

About user exits

MQSeries link for R/3 provides a user exit facility that you can use to provide entry points to your own software routines. When enabled, the relevant exit handler calls the user exit for each MQSeries message that passes through the server.

What any user exit does depends on your requirements. All you need to do is write a suitable program for the function that you need, adhering to the rules described here. In particular, you must use the supplied return codes to tell the exit handler what to do next. For example, if a certain operation fails, and you want the user exit to end, you must issue a return code of `SMQRC_TERMINATE_EXIT`.

MQSeries link for R/3 provides the source for two sample user exits, written in C, that you can modify to provide entry points to your own software routines. See “Sample user exits” on page 53 for more information.

When the exits are called

Calling the exits differs on the outbound and inbound servers.

User exits on the outbound server

If the exit is enabled, when the outbound server is started it calls the Initialise entry point, which sets up the exit. After this, R/3 passes control to the exit. The first time the exit is called, the Initialise function is invoked to set up the exit. After this, the Execute and Return functions are called as before.

User exits

There can be multiple exits active on the outbound server. If an exit is specified in the ini file the Initialise function is called when the server is started. If an exit is specified for a particular destination and the *Call User Exit* flag is set in the **smqDestConf** file, the Initialise function is called when the first data is received from R/3. After initialization, the execute function is called whenever data is received for a destination with an exit specified or implied before the MQSeries put to the output is attempted. The return function is called after the put if the user exit indicates that the message has been modified.

Figure 8 shows the outbound user exit being called before an MQSeries message is put onto the outbound message queue.

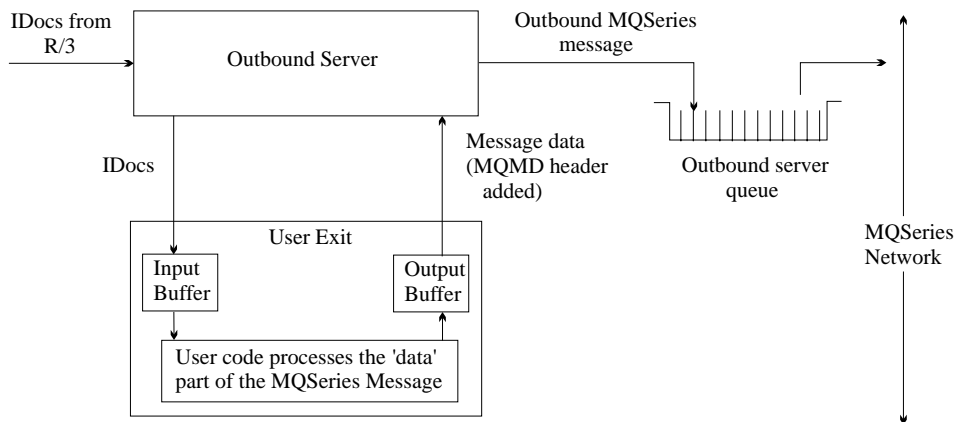


Figure 8. Data flow through the outbound server user exit

User exits on the inbound server

If the exit is enabled, when the inbound server is started it calls the Initialise entry point, which sets up the exit. After this, whenever a message is got from the inbound message queue, the server calls the Execute entry point. After the data has been transferred to R/3, the server calls the Return entry point if the user exit indicates that the message has been modified. Finally, when the server ends, it calls the terminate entry point.

Note: The execute and return functions are called from within an MQSeries unit of work (UOW). The retrieval of the message from the inbound queue is not committed until after the return function is completed. Initialise and terminate are called outside of an existing UOW.

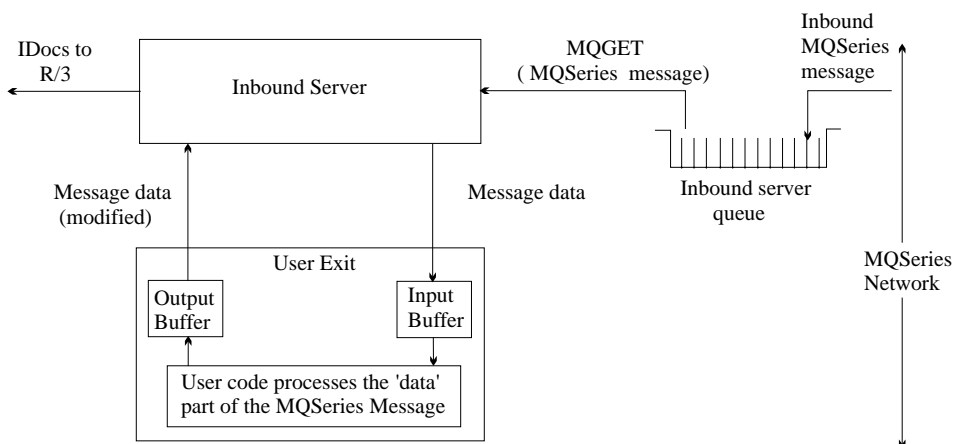


Figure 9. Data flow through the inbound server user exit after initialization

Enabling the user exits

The method you use to enable a user exit depends on the type of exit.

Enabling user exits for the outbound server

For the outbound server, you enable the exit by specifying values in the **smqDestConf** file. See “Task 5. Mapping R/3 Logical Systems to MQSeries destinations” on page 35 for details.

There is an optional exit buffer that you can use to pass your own user-defined data to the exit. You can specify this data using the *exitbuffer* parameter in the outbound **ini** file.

Note: If you want the outbound server to use the *exitname* and *exitbuffer* parameters in the outbound **ini** file, specify an asterisk (*) in the *exitname* field and set the *calluserexit* to 'yes' in the **smqDestConf** file.

Enabling user exits for the inbound server

For the inbound server, you enable the exit by specifying `invokeexit=y` in the initialization file that you specify on the `smqsi` command. You must also specify a path and name for the executable. For example, if your initialization file for the inbound server contains these lines:

```
invokeexit=y
exitname=/home/your_name/exits/dothis.a
```

the exit **dothis.a**, in directory `/home/your_name/exits`, is called after the exit handler retrieves each MQSeries message from the inbound server input queue.

User exits

Note: On OS/400, the value of the *exitname* parameter is the name of the service program, which can be qualified with the name of a library. For example:

```
    exitname=dothis  
or  
    exitname=mylib/dothis
```

There is an optional exit buffer that you can use to pass your own user-defined data to the exit. You can specify this data using the *exitbuffer* parameter in the inbound **ini** file.

Understanding the external interface

The exit handler uses a defined set of call interfaces to communicate with the user exit. There are four entry points:

Initialise

Sets up the user exit. This must be the first, or only, entry point 'exported' when linking the exit.

Execute

Typically, processes data.

Return

Cleans up any resources required by the Execute function.

Terminate

Terminates the user exit when the processing is complete.

Note: The entry point names are arbitrary; you do not have to use the names above although you will see them used in the samples.

The interface parameters are structures rather than individual values. Definitions of the data structures and the call interface are in the supplied include file **smqc.h**.

You must write all the user exit software in C and run it in the same process as the exit handler. Any cross-process calls and data sharing required by the user exit must be managed by the user exit software.

Recommendation

For complete information about the user exit, refer to the supplied sample user exit, which is located in the **samples** sub-directory.

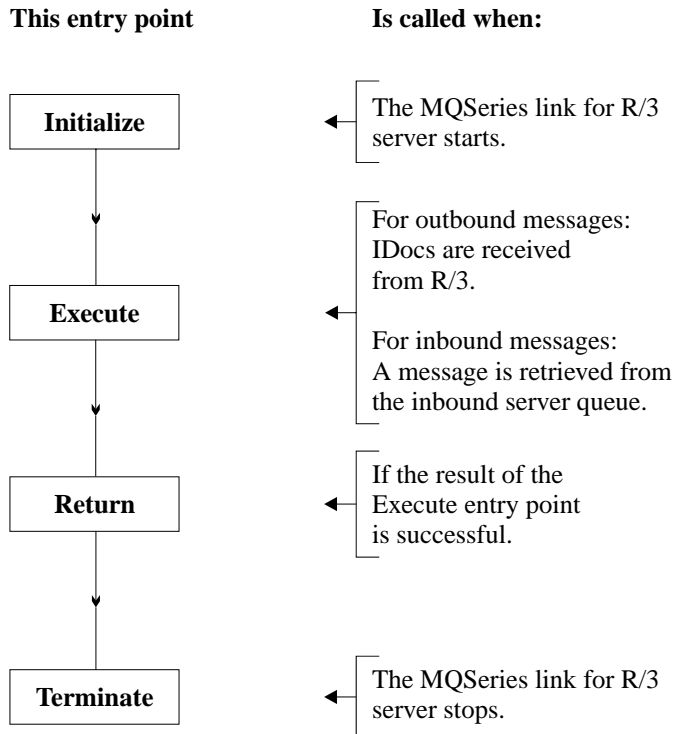


Figure 10. Entry points in a user exit. Initialise is the first entry point called.

The Initialise entry point (`smqUserExitInitialise`)

The first entry point in your code must be of type `smqUserExitInitialize`.

This entry point enables the exit handler to pass setup information to the user exit. It also enables the user exit to perform its initialization procedures and to pass the addresses of the other entry points back to the exit handler. The server calls this entry point with two parameters:

SMQEXIT_SETUP

This structure pointer is passed to all user exit calls. It contains several fields that are reserved for user exit use. They are:

- `ExitHandle`
- `ExitOptions`
- `Handler Options`

They are all defined as `MQLONG` and may contain pointers to user exit data and structures that are required to be available to more than one entry point.

User exits

SMQEXIT_INIT

This is a pointer to a structure containing fields for the addresses of the remaining entry points and result code. Your user exit must supply the data for these fields:

Execute

Address of the execution entry point

Return

Address of the return entry point

Terminate

Address of the termination entry point

Execute entry point (smqUserExitExecute)

This is the call interface that executes the conversion procedures within your user exit. The server calls this entry point with three parameters:

SMQEXIT_SETUP

A pointer to a structure used with the call to the Initialise procedure.

SMQEXIT_INPUT

Pointers to a structure containing information about the data to be converted and a pointer to the data (the input buffer). Your code must not alter the fields in this structure.

The pointers vary according to the direction of the message, inbound or outbound.

Header

On inbound this will be NULL. On outbound this is a pointer to the MQSAPH structure as generated by the server.

MessageDesc

On inbound, a pointer to the MQMD associated with the retrieved message. On outbound, a pointer to the MQMD that has been generated by the server.

MessageData

On inbound, a pointer to the message as retrieved from the inbound server queue. On outbound, a pointer to the IDocs as received from R/3.

MessageDataLen

On inbound, the length of the message as retrieved from MQSeries. On outbound, the length of the IDoc data as received from R/3.

Note: This does not include the length of the MQSAPH.

SMQEXIT_OUTPUT

A set of pointers including, for a successful conversion:

Header

On outbound, this should be NULL. On inbound, this points to the MQSAPH structure generated by the exit.

MessageData

On outbound, this points to the message data generated by IDocs that are generated by the exit. This data is placed on the outbound queue. On inbound, this points to the IDocs that are generated by the exit.

MessageDataLen

For both inbound and outbound, the length of the data addressed by MessageData.

Result A return code indicating the result of the processing by your code. The return codes that you can use assume a data conversion operation (see “Return codes for the execution entry point”).

Return codes for the execution entry point

Depending on the results of your processing, you must specify one of the following return codes:

SMQ_CONVERT_OK

The conversion was completed without error.

The contents of the output buffer (that is, the results of processing by the exit) are inserted into an MQSeries message, which is then put on the appropriate server queue.

SMQ_CONVERT_NOT_NEEDED

Your code decided that the proposed conversion was not required.

The server will then act as though the exit was not called.

SMQ_CONVERT_FAIL

The proposed conversion was not completed successfully. The message will be put to the bad message queue if one has been specified.

Note: This may cause the server to end.

SMQ_TERMINATE_EXIT

The proposed conversion was not completed successfully, and you want to terminate the exit.

SMQ_TERMINATE_SERVER

The proposed conversion was not completed successfully, and you want to terminate this instance of the server.

User exits

Return entry point (smqUserExitReturn)

This entry point is called only if the result of the preceding execute procedure was successful.

When the exit handler has completed processing the data returned from the execution call, it calls the return entry point. The return enables the user exit to perform any cleanup, such as freeing memory.

The server calls this entry point with two parameters:

SMQEXIT_SETUP

A pointer to a structure that was used with the call to the Initialise procedure.

SMQEXIT_OUTPUT

A pointer to a structure that was returned by the previous execution procedure. This structure also contains pointers to the memory allocated to contain the converted data in the user exit, the length of the converted data, in bytes, and a return code indicating the result of the data conversion.

Return codes for smqUserExitReturn

Depending on the results of your processing, you must specify one of the following return codes:

SMQRC_OK
SMQRC_TERMINATE_EXIT
SMQRC_TERMINATE_SERVER

The following return codes are not valid for this entry point:

SMQ_CONVERT_OK
SMQ_CONVERT_NOT_NEEDED
SMQRC_CONVERT_FAIL

Your code should not generate these return codes for this entry point.

Terminate entry point (smqUserExitTerminate)

The termination procedure is called when the server ends either normally or because of an error condition. The exit handler calls the termination procedure in the user exit to perform any cleanup, and terminate gracefully. The user exit must complete termination before returning from this call. The exit handler unloads the user exit when this call completes.

The server calls the termination entry point with two parameters:

SMQEXIT_SETUP

A pointer to a structure that was used with the call to the Initialise procedure.

SMQEXIT_REQUEST

An enumerated type indicating the reason for the termination:

SMQ_REQUEST_EXIT

The exit returned SMQRC_TERMINATE_EXIT.

SMQ_REQUEST_SERVER

The exit should end because the server is terminating.

SMQ_REQUEST_ERROR

The exit should end because the server is terminating with an error.

Sample user exits

Two user exits are supplied that you can use as a basis for your own exits.

smqesmp1.c

Shows the exit structure

smqesmp2.c

shows sending data to a non-R/3 system or receiving data from an R/3 system.

The source code for these exits is supplied in the **samples** directory,

Note: On OS/400 the exit code source is in file C in library **QMQLINK**.

See “Sample user exits” on page 99 for more information about the samples.

Compiling user exits

The exits are dynamically loaded objects; they must have a name of no more than eight non-blank characters. For example:

```
MYEXIT
```

Note: The MQSAPH header structure, which must be at the start of all MQSeries link for R/3 messages, should be packed (aligned on 1-byte boundaries). The include file `smqc.h` defines a way to do this for each platform. To activate the correct alignment of this structure, you must define a compiler variable appropriate to your platform. The platforms and compiler variables are:

Compiling user exits

Platform	Variable
AIX	SMQ_AIX
HP-UX	SMQ_HPUX
SunOS	SMQ_SOLARIS
OS/400	SMQ_OS400
Windows NT	SMQ_NT

The following sample commands show how to specify these compiler variables.

For AIX:

```
cc -c -I<Include> smqesmp1.c -DSMQ_AIX
ld exit.o -e Initialise -o MYEXIT -bM:SRE -H512 -T512 -L <Libraries> -lc
```

For Compaq Tru64 UNIX:

```
cc -c -std1 -I<Include> exit.c
ld -shared exit.o -o MYEXIT -lc
```

Note: If you use MQI calls in your exits (for example, as in the smqesmp2.c sample), you need to link to the MQSeries libraries.

For HP-UX:

```
cc -c -Aa +z -I<Include> smqesmp1.c -DSMQ_HPUX
ld -b exit.o -o MYEXIT +I Initialise -L <Libraries>
```

For SunOS:

```
cc -c -KPIC -I<Include> smqesmp1.c -DSMQ_SOLARIS
ld -G exit.o -o MYEXIT
```

Note: For Solaris, the entry point name *must* be 'Initialise'.

For OS/390:

For smqesmp1.c:

```
cc -e Initialise -o smqesmp1 \
-I'/'thlqual.SCSQC370" -I/usr/lpp/smq/include \
-W c,dll,expo -W l,dll, smqesmp1.c
```

For smqesmp2.c:

```
cc -e Initialise -o smqesmp2 \
-I'/'thlqual.SCSQC370" -I/usr/lpp/smq/include \
-W c,dll,expo -W l,dll, smqesmp2.c \
-lm "'/'thlqual.SCSQLOAD(CSQBSTUB)'"
```

where thlqual is the high level qualifier of the MVS data set for MQSeries.

For OS/400

```
CRTCMOD MODULE(MYLIB/MYEXIT) SRCFILE(MYLIB/MYFILE) SYSIFCOPT(*ALL)
CRTSRVPGM SRVPGM(MYLIB/MYEXIT) MODULE(MYLIB/MYEXIT) EXPORT(*ALL)
```

For Windows NT:

```
cl /nologo /W3 /GX /Zi /YX /Od /D "WIN32" smqesmp1.c
/D "SMQ_NT" /link /nologo /SUBSYSTEM:console
/INCREMENTAL:yes /PDB:"MYEXIT.PDB"
/MACHINE:I386 /OUT:MYEXIT.DLL /EXPORT:Initialise,@1 /DLL
```

Notes:

1. The samples assume a name of 'Initialise' for the Initialise entry point.
2. In the Windows NT environment the initialise entry point must be the first entry point in the DLL and not just in the source file.

Message formats

To send transactions to or from R/3, MQSeries link for R/3 uses MQSeries messages, that is, messages that have the same format as those used in MQSeries. Figure 11 shows the structure of a message, broken down into its constituent levels.

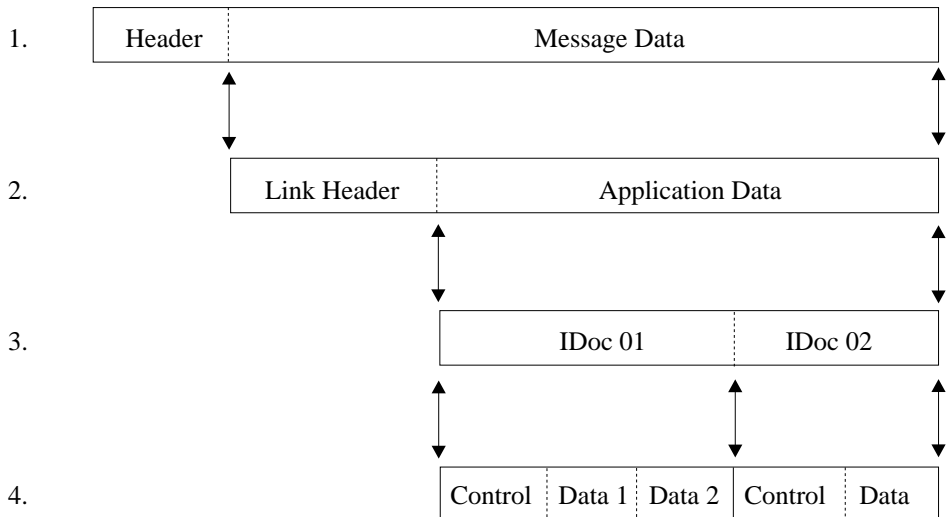


Figure 11. MQSeries message structure and IDocs

1. This is the generic MQSeries message format. The header contains MQSeries control information for processing the message, and is followed by message data.
2. The message data begins with a link header that is used by the MQSeries link for R/3 software (and in some cases by non-SAP applications). The

Message formats

link header contains information to identify the destination of the message. Figure 12 on page 57 shows the details of the MQSeries link for R/3 header structure.

The link header is followed by application data. Typically, for inbound transactions, the application data field is in IDoc format.

For outbound transactions the application data field should be converted to the format required by the target application. However, if the target application is another R/3 system, this field takes an IDoc format.

3. In an MQSeries message, the application data consists of one or more IDocs stored contiguously with no separators between them. The maximum size of the application data is specified in the **ini** file, subject to a maximum of 100 MB for MQSeries Version 5 messages (4 MB for MQSeries Version 2 messages). See “Other defaults for the inbound server” on page 79 for more information.
4. Each IDoc has a control table and one or more data tables of fixed length. The structure details depend on which version of SAP R/3 is generating the IDocs:

Version 3 IDocs

These use the following structures

EDI_DC This control structure size is 464

EDI_DD This data structure size is 1055

Version 4 IDocs

These use the following structures

EDI_DC40 This control structure size is 524

EDI_DD40 This data structure size is 1063

Note: For a Version 4 IDoc the **TABNAM** field in the EDI_DC40 structure is of length 10 characters and has the value 'EDI_DC40 ' (that is two spaces after the 0)

Each IDoc is identified by a unique IDoc number that is stored in a field in both the control and data tables. A new IDoc within the same MQSeries message is identified by a new IDoc number contained in the control table. Figure 11 on page 55 shows a message with two IDocs. The first has two data parts and the second has one data part.

MQSeries link for R/3 header structure

The MQSeries link for R/3 header contains information about the message, including information about the remote R/3 destination of the message. The outbound server automatically inserts an MQSeries link for R/3 header in every outbound message. If an application is sending a message inbound R/3 using MQSeries link for R/3, the sending application must construct this header.

Link Header	Application Data		
	Field Name	Length	Type
	StrucId	4	Required ¹
	Version	4	Required ¹
	StrucLength	4	Required ²
	Encoding	4	Optional ⁴
	CodedCharSetId	4	Optional ⁴
	Format	8	Optional ⁴
	Flags	4	Reserved
	Client	3	Optional ³
	Language	1	Optional ³
	HostName	48	Optional ³
	UserId	12	Optional ³
	Password	8	Optional ³
	SystemNumber	2	Optional ³
	Reserved	2	Optional
	Total length of structure	108	See Note 2

Figure 12. Structure of the MQSeries link for R/3 header

Notes:

1. *StrucId* and *Version* are required fields.
 - The structure identifier, is 'SAPH'.
 - The version number is the value '1' in a long integer field.
2. The total structure length is required even though some field values are optional. *StrucLength* always has the value 108.
3. For inbound messages, optional values that are set in this header override those set when the inbound server was initialised from the **ini** file.
For outbound messages, any of these values that are set in the **smqDestConf** file are used to complete these fields in this header.
4. When you use MQSeries link for R/3 with MQSeries Version 5.1 and above, you should set the *Format* field to MQSTR to describe the IDoc data that follows the MQHSAP header. You should also set the *Encoding* and *CodedCharSetId* fields.

Message formats

Constructing messages with IDocs

If you send data to an inbound server from a source outside R/3, you must ensure that the MQSeries message data conforms to, or is converted to IDoc format, with control and data sections, as shown in Figure 11 on page 55.

If you have more than one IDoc in each message, you must ensure that each IDoc has a 16-byte unique identifier in the *Docnum* field which starts at offset 13 of the control section of each IDoc. Otherwise, the inbound server treats the whole package as one IDoc and reports error message SMQ4172 - Invalid length for IDoc.

If your IDocs originate from R/3, you should specify the identifiers in the **smqDestConf** file.

Code page conversion is required if you are sending messages to the inbound queue from a platform with a different code page. The inbound server uses the MQSeries MQGET call option, MQGMO_CONVERT, and automatically calls the supplied data conversion exit **MQHSAP** if the message has the format field in the MQMD structure set to 'MQHSAP'. The outbound server sets this automatically, but if you use your own program to put messages to the inbound queue, you need to set the format field in the MQMD structure to MQHSAP (or the name of your data conversion exit if you are using your own) when you issue the MQPUT call. When you use MQSeries link for R/3 with MQSeries Version 5.1 or later, you should set the *Format* field in the MQHSAP header to MQSTR to describe the IDoc data that follows the MQHSAP header.

See the chapter on user exit programs in the *MQSeries Intercommunication, SC33-1872* for more information about data conversion exits.

Note: When compiling your own programs to construct messages that are to be transferred from MQSeries to R/3, you must ensure that the MQSAPH header structure is packed (aligned on 1-byte boundaries).

Chapter 7. Command reference

This chapter describes the commands that you can use to start either the inbound server or outbound server. (See also “Chapter 8. Running servers as Windows NT services” on page 67.)

smqsi Start the inbound server; see page 60.

smqso Start the outbound server; see page 62.

Initialization parameters

When you start a server, it reads the command parameters and the contents of the initialization (**ini**) file specified by the *-i* parameter on the command line. If the same parameters are defined on both the command line and in the **ini** file, the command line parameters are the ones that are actually used.

As far as possible, use the command line to specify particular values of parameters; use the **ini** files for frequently-used values.

smqsi (Start inbound server)

Purpose

On Unix systems, if you start the server from a directory other than **smq/bin**, ensure that your starting directory has the same permission as the **smq/bin** directory.

Use the `smqsi` command to start the MQSeries link for R/3 inbound server, allowing information to flow inbound into an R/3 system from a specified MQSeries queue.

When you issue this command, it must contain a `-iInitializationFileName` parameter, or the server will not start. If load balancing is to be used on the inbound server, the **saprfc.ini** file must be in the working directory from which the server is started.

Command line parameters take precedence over any parameters you specify in the **ini** file.

Format

```
smqsi -iInitializationFileName  
      -mQueueManagerName  
      -qInboundQueueName
```

Note: On OS/400, the entire parameter string must be enclosed in single quotes.

Parameters

`-iInitializationFileName`

Specifies the full path and file name of the initialization file. This file contains parameters that customize the execution of the inbound server.

See “Inbound server initialization parameters” on page 75 for more information.

`-mQueueManagerName`

Specifies the name of the MQSeries queue manager that owns the MQSeries objects specified for the inbound server. If specified, this parameter overrides the `QueueManagerName` parameter, if any, specified in the inbound **ini** file. If you do not specify a queue manager in any of these places, the default queue manager on the host running the inbound server is used.

The specified queue manager must be running, or the server cannot start.

-q*InboundQueueName*

Specifies the queue name used to store messages being transferred to the R/3 system. This queue must have already been created before the start command is called.

You must specify a valid inbound queue name on either the command line or in the **ini** file, or the server cannot start.

Examples

1. Start the inbound server and take all initialization data from the **myini.ini** file.

```
smqsi -imyini.ini
```

On OS/400:

```
smqsi '-imyini.ini'
```

2. Start the inbound server, reading the initialization data from the specified file **in.ini**, except for the inbound queue, which is to be **inbound.queue**. If the **ini** file has an entry for the *InboundQueueName* parameter, this is ignored.

```
smqsi -id:\saplink\in.ini -qinbound.queue
```

On OS/400:

```
smqsi '-i/saplink/in.ini -qinbound.queue'
```

Note: Remember to stop the inbound server when taking R/3 down.

Running the inbound server as a batch process on OS/400

Use the SBMJOB command to run the inbound server as a batch process on OS/400. For example:

```
SBMJOB CMD(SMQSI PARMLIST('-i/myini.ini')) JOB(SMQSI) JOBQ(QSYSNOMAX)
```

Running the inbound server using the AS/400 Qshell interpreter

If you are running on AS/400 V4R2 or later you can run the inbound server using the Qshell interpreter. One of the advantages of using the Qshell interpreter is that quotes are not needed around the parameter string. This means you can start the inbound server the same way as it is started in the other MQSeries link environments.

To run the inbound server using the Qshell interpreter perform the following steps:

1. Add a symbolic link to the inbound server program using the following command:

```
ADDLNK OBJ('/qsys.lib/qmqlink.lib/smqsi.pgm')
NEWLNK('/usr/bin/smqsi') LNKTYP(*SYMBOLIC)
```

2. Start the shell environment using the following command:

smqsi

qsh

3. Start the inbound server. The following is an example command:

```
smqsi -i/myini.ini
```

For more information about using the AS/400 Qshell interpreter see the online documentation available on the internet at the following url:

<http://publib.boulder.ibm.com/pubs/html/as400/ic2924/info/java/rzahz/qshell.htm>

Stopping the server

To stop the inbound server, type CTRL+C.

Notes:

1. On OS/390, use the following command in UNIX System Services to send a SIGINT signal to the MQSeries link for R/3:

```
kill -2 pid
```

where pid is the process id of the link you want to stop.

You can also use JCL to stop the inbound server. See “OS/390 samples” on page 103 for an example of JCL to stop the outbound server. You must modify this example if you want to stop the inbound server.

2. On OS/400, use System-Request option 2 (cancel) to stop the inbound server from an interactive session.

smqso (Start outbound server)

Purpose

On Unix systems, if you start the server from a directory other than **smq/bin**, ensure that your starting directory has the same permission as the **smq/bin** directory.

Use the **smqso** command to start the MQSeries link for R/3 outbound server. The outbound server takes one or more IDocs from an R/3 system, converts them to an MQSeries message, and puts this message on the (MQSeries) outbound message queue specified by the required destination.

To start the outbound server, the *gateway service*, *gateway host*, and *program Id*, must be defined at least once in the **ini** file or on the command line. The **smqDestConf** file must also be present in the working directory from which the server is started.

If you specify a parameter in both the **ini** file and on the command line, the command line parameters take precedence.

Format

```
smqso -iInitializationFileName
      -xGatewayService
      -gGatewayHost
      -aProgramId
      -mQueueManagerName
```

Note: On OS/400, the entire parameter string must be enclosed in single quotes.

Parameters

-iInitializationFileName

Specifies the full path and file name of the initialization (ini) file for the outbound server. This file contains parameters to customize the execution of the outbound server. See “Outbound server initialization parameters” on page 71 for more information.

-xGatewayService

Specifies the gateway service used by the R/3 connection. You must specify a valid gateway service either on the command line or in the **ini** file. The name of the gateway service used with the command must match the name specified in the R/3 RFC destination panel. Otherwise, the outbound server cannot complete its startup procedure.

-gGatewayHost

Specifies the gateway host used by the R/3 connection. You must specify a valid gateway host either on the command line or in the **ini** file. The name of the gateway service used with the command must match the name specified in the R/3 RFC destination panel. Otherwise, the outbound server cannot complete its startup procedure.

-aProgramId

A unique identifier that specifies which host is to receive the outgoing R/3 information. The name of the program ID used with the command must match the name specified in the R/3 RFC destination panel. Otherwise, the outbound server cannot complete its startup procedure.

-mQueueManagerName

Specifies the name of the MQSeries queue manager that owns the MQSeries objects used by the outbound server. This parameter overrides the *queuemanagername* parameter, if any, specified in the outbound **ini** file.

If you do not specify a queue manager name on either the command line or in the **ini** file, the default queue manager on the host running the outbound server is used.

The specified queue manager must be running, or the server cannot start.

Examples

1. Start the outbound server, taking all its initialization data from the **ini** file **out.ini**:

```
smqso -id:\saplink\out.ini
```

On OS/400:

```
smqso '-i/saplink/out.ini'
```

2. Start the outbound server reading the initialization data from the **ini** file **out.ini**. Then override the *ProgramId*, *GatewayHost*, and *GatewayService* parameters in the file with those specified on the command line below. This allows the server to connect to a different preconfigured R/3 RFC destination containing, in this case, the parameters *prog_id_1*, *Host_1*, and *sapgw45*.

```
smqso -aprog_id_1 -gHost_1 -xsapgw45 -id:\saplink\out.ini
```

On OS/400:

```
smqso '-aprog_id_1 -gHost_1 -xsapgw45 -i/saplink/out.ini'
```

3. Start the server, reading all the initialization data from the specified file and then override the *QueueManagerName* parameter. This server uses 'saturn.queue.manager' as the destination queue manager.

```
smqso -id:\saplink\myini.ini -msaturn.queue.manager
```

On OS/400:

```
smqso '-i/saplink/myini.ini -msaturn.queue.manager'
```

Running the outbound server as a batch process on OS/400

Use the SBMJOB command to run the outbound server as a batch process on OS/400. For example:

```
SBMJOB CMD(SMQSO PARMLIST('-i/out.ini')) JOB(SMQSO) JOBQ(QSYSNOMAX)
```

Running the outbound server using the AS/400 Qshell interpreter

If you are running on AS/400 V4R2 or later you can run the outbound server using the Qshell interpreter. One of the advantages of using the Qshell interpreter is that quotes are not needed around the parameter string. This means you can start the outbound server the same way as it is started in the other MQSeries link environments.

To run the outbound server using the Qshell interpreter perform the following steps:

1. Add a symbolic link to the outbound server program using the following command:

```
ADDLNK OBJ('/qsys.lib/qmqlink.lib/smqso.pgm')  
NEWLNK('/usr/bin/smqso') LNKTYPE(*SYMBOLIC)
```


2. Start the shell environment using the following command:

```
qsh
```

3. Start the inbound server. The following is an example command:

```
smqso -iout.ini
```

For more information about using the AS/400 Qshell interpreter see the online documentation available on the internet at the following url:

<http://publib.boulder.ibm.com/pubs/html/as400/ic2924/info/java/rzahz/qshell.htm>

Stopping the server

To stop the outbound server, type CTRL+C.

Notes:

1. On OS/390, use the following command in UNIX System Services to send a SIGINT signal to the MQSeries link for R/3:

```
kill -2 pid
```

where pid is the process id of the link you want to stop.

You can also use JCL to stop the outbound server. See “OS/390 samples” on page 103 for an example of JCL to stop the outbound server.

2. On OS/400, use System-Request option 2 (cancel) to stop the outbound server from an interactive session.

smqso

Chapter 8. Running servers as Windows NT services

The outbound and inbound server programs, **smqso.exe** and **smqsi.exe**, can be run as Windows NT Services as well as from the command line (see “Chapter 7. Command reference” on page 59).

Note: If you want to pass arguments unique to this invocation of the servers, you must use the command line method.

To run the outbound or inbound server as a service, you need to run the additional **srvsetup.exe** program which is provided with MQSeries. This program adds the command line options required when running the servers as services.

To run the server setup for the outbound server, enter:

```
srvsetup -m QM
        -x G/Way
        -a ProgID
        -g GW/Host
        -i IniFile
        -p PathName
        -o
        -d
```

To run the server setup for the inbound server, enter:

```
srvsetup -q Q
        -m QM
        -i IniFile
        -p PathName
        -n
        -d
```

where:

- p** is the absolute path to the outbound server program, **smqso.exe** or the inbound server program, **smqsi.exe**. This parameter is required.
- o** indicates that the outbound server program is being registered.
- n** indicates that the inbound server program is being registered.
- d** provides optional debug output so that some diagnostics are displayed when the arguments are written to the registry.

The other parameters are described in “Chapter 7. Command reference” on page 59.

Windows NT services

To get the Link services to wait for the MQSeries queue manager to start up, you must add a number of values to the system registry. Use **regedt32.exe** to edit the system registry.

Under the key:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services

is one, or both of, two other keys:

MQSeriesSAPLinkInBoundServer

MQSeriesSAPLinkOutBoundServer

depending on which services have been set up.

Under each of these add a new value:

Value Name: DependOnService

Value Type: REG_MULTI_SZ

Value Data: MQSeriesServices

and reboot the system.

Changes to the Windows NT system registry

When the server setup program is run, the following keys are generated in the registry:

Outbound Server

HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeriesSAPLinkData\CurrentVersion\Out\

szOutBoundServerPath	- d:\smq\bin
szOutExeName	- smqso.exe
szOutFileName	- out.ini
szOutGatewayHost	- 9.20.23.91
szOutGatewayService	- sapgw00
szOutProgID	- TESTProg_ID
szOutQueueManager	- SAPLINKQM

Inbound Server

HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeriesSAPLinkData\CurrentVersion\In\

szInBoundServerPath	- d:\smq\bin
szInExeName	- smqsi.exe
szInFileName	- in.ini
szInQueueManager	- SAPLINKQM
szInQueueName	- SAPLINKQ

The Services are registered in:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\  
MQSeriesSAPLinkInBoundServer  
MQSeriesSAPLinkOutBoundServer
```

When you have run `srvsetup` to register the servers, reboot the machine to enable the outbound and inbound Server services to run.

Once the machine has been re-booted, you can navigate from the Start menu, (following the path Start\Settings\Control Panel\Services), to see the entries for the IBM MQSeries SAPLink OutboundServer and InboundServer.

These services can be manipulated in the same way as any other service, for example, they may be started and stopped from the Service Control Manager.

Note: 'Pause' is not supported.

Removing the Out and In Bound Servers from the Service Control Manager

To remove the both the outbound and inbound SAPLink services, run the program, **`svrclean.exe`**. This program removes all the entries added to the registry for the command line arguments used for the servers as well as the entries that install the servers as services.

When the program is running, press 'Q' to quit running the program or press 'G' to proceed with running the program.

Once the **`svrclean`** program has been run the machine must be re-booted before attempting to re-install the SAPLink services.

Chapter 9. Initialization files

The chapter tells you about the initialization (**ini**) files that are used when you start one or other of the MQSeries link for R/3 servers. Using an **ini** file is a convenient way to specify parameters that you use repeatedly. Also, using the default definitions minimizes the time you need to spend on configuration.

There are two different types of **ini** file, one for each server type. For the outbound server, specifying an **ini** file on the `smqso` command is optional. When you start the inbound server, you *must* specify the name of the **ini** file using the `-i` option on the `smqsi` command. For more information about these commands, see “Chapter 7. Command reference” on page 59.

Outbound server initialization parameters

You can create an outbound initialization file to specify parameters that are used when you start the outbound server (using the `smqso` command).

Note: Any parameters you specify in this file are overridden if you specify the same parameters on the `smqso` command.

For the outbound server, there are four categories of parameters:

1. RFC destination parameters
2. MQSeries definitions
3. User exit information
4. Other defaults

RFC destination parameters

The server can establish a connection with an RFC destination, but only if the RFC Destination parameters match the following three parameters.

Parameter	gatewayhost
Example	<code>gatewayhost=ha0016</code>
Description	Server name or IP address where gateway service for SAP is running
Maximum Length	48 characters
Format	IP hostname or IP address

Outbound ini files

Parameter	gatewayservice
Example	gatewayservice=sapgw86
Description	Name of the R/3 gateway service residing on the named gateway host.
Maximum length	12 characters
Format	sapgwXX, where 'XX' is the system instance on which the gateway service is running

Parameter	programid
Example	programid=telstaret
Description	Name of the unique program ID parameter.
Maximum length	24 characters
Format	Conforms to R/3 naming conventions.

Outbound user exit defaults

These defaults are used when you type an asterisk (*) in the **Exit Name** field in the **smqDestConf** file.

Parameter	exitname
Examples	exitname=d:\saplink\bin\smqesmp1.dll exitname=/usr/ableman/saplinkbin/smqesmp1.a
Description	The name of the user exit DLL or shared library to be used by the exit handler. If you do not specify a path here, you must specify one in the PATH environment variable for your system. This exit name and path are used if you specify an asterisk (*) in the Exitname field and set <i>Call User Exit</i> to 'yes' in the smqDestConf file.
Maximum length	40 characters
Format	Conforms to operating system format for path and file name.

Parameter	exitbuffer
Example	exitbuffer=LosAndos
Description	Specifies the optional contents of a buffer passed to the user exit on initialization. You can use this parameter to define a switch to give different user exit functions depending on these contents.
Maximum length	32 characters
Format	Free form text.

MQSeries definitions

The following parameters define the names of the queue manager and the transaction queue.

Note: The outbound message queue name is defined only in the **smqDestConf** file

Parameter	transactionqueue
Example	transactionqueue=orange.local.queue
Description	Specifies the name of the queue used by the outbound server to store transaction ids. If you do not specify this parameter, the queue name SMQ_OUTBOUND_TRANIDS, as defined in outbound server code, is used as the default.
Maximum length	48 characters
Format	Conforms to the MQSeries queue name specifications.

Parameter	queuemanager
Examples	queuemanager=MN01 queuemanager=saturn.queue.manger
Description	Specifies the name of the local queue manager that owns the transaction id queue objects. (This may also be, but not necessarily, the same queue manager that owns outbound server queue.) If you do not specify a name here, the local default queue manager is used if there is one.
Maximum length	48 characters
Format	Conforms to the MQSeries queue manager name specifications.

Other defaults for the outbound server

The following parameters specify other defaults used by the outbound server.

Outbound ini files

Parameter	rfctraceon
Examples	rfctraceon=yes rfctraceon=y rfctraceon=no rfctraceon=n
Description	Option to turn trace on or off. The trace file is saved in dev_rfc in the current directory. By default, tracing is switched off.
Maximum length	3 characters
Format	y, n, yes, no

Parameter	maxidocsize
Example	maxidocsize=500,000
Description	Specifies the maximum size of an outbound IDoc batch. The default is 2,097,152 bytes.
Maximum length	9 characters
Format	numeric

Parameter	batching
Examples	batching=yes batching=y batching=no batching=n
Description	Allows Idocs that are received in a batch to be placed in a single MQSeries message.
Maximum length	3 characters
Format	y, n, yes, no

Parameter	logging
Examples	logging=yes logging=y logging=no logging=n
Description	Controls the writing of status messages to the error log.
Maximum length	3 characters
Format	y, n, yes, no

Inbound server initialization parameters

You can create an inbound initialization (**ini**) file to specify parameters that are used when you start the inbound server (using the `smqs i` command).

Note: Any parameters you specify in the **ini** file are overridden if you specify the same parameters on the command line.

For the inbound server, there are four categories of parameters:

1. Remote R/3 system connection parameters
2. MQSeries definitions used by the inbound server
3. User exit information
4. Other defaults

R/3 system connection defaults

The following parameters specify the defaults for the destination information of inbound messages. The inbound server tries to establish a connection on receiving an inbound message using these connection variables.

Parameter	client
Example	client=100
Description	Specifies the client number of the R/3 system for inbound server connection; that is, the destination of the messages.
Maximum length	3 characters
Format	Conforms to R/3 naming conventions; always a number.

Parameter	user
Example	user=fred
Description	Specifies the R/3 logon user name for an inbound server connection to the remote R/3 system
Maximum length	12 characters
Format	Conforms to R/3 naming conventions.

Parameter	language
Example	language=e
Description	Specifies the language of the R/3 system for inbound server connection to the remote R/3 system
Maximum length	1 byte
Format	Conforms to R/3 naming conventions: always a letter.

Inbound ini files

Parameter	password
Example	password=leisa
Description	Specifies the R/3 user password for the inbound server connection to the remote R/3 system
Maximum length	8 characters
Format	Conforms to R/3 naming conventions.

Parameter	sysnbr
Example	sysnbr=00
Description	Specifies the instance number of the R/3 system for inbound server connection to the remote R/3 system.
Maximum length	2 characters
Format	Conforms to R/3 naming conventions.

Parameter	hostname
Examples	hostname=9.165.255.88 hostname=abletaman
Description	The application server name used by the R/3 system for an inbound server connection to the remote R/3 system
Maximum length	48 characters
Format	Conforms to R/3 naming conventions.

MQSeries definitions used by the inbound server

These parameters specify the name of the MQSeries queue manager and the queues to be used with the inbound server. In addition, one of the parameters specifies whether the server is to be stopped if it receives a message in a format unrecognizable to R/3. (For more information about unrecognizable messages, see “Handling unrecognizable messages” on page 86.)

Parameter	transactionqueue
Examples	transactionqueue=red.local.queue transactionqueue=SMQ_INBOUND_TRANIDS
Description	Specifies the name of the queue used by the inbound server to store transaction ids. The default, if you do not specify anything, is specified as SMQ_INBOUND_TRANIDS.
Maximum length	48 characters
Format	Conforms to the MQSeries queue name specifications.

Parameter	queuemanager
Example	queuemanager=saturn.queue.manager queuemanager=ehningen.qmgr
Description	Specifies the name of the queue manager that owns the inbound queue object (and the inbound transaction queue). If you do not specify a name here, the local default queue manager is used, if one exists.
Maximum length	48 characters
Format	Conforms to the MQSeries queue manager name specifications.

Parameter	terminateonbadmessage
Examples	terminateonbadmessage=y terminateonbadmessage=yes terminateonbadmessage=n terminateonbadmessage=no
Description	Specifies whether the inbound server is stopped if it receives a message that is unrecognizable. The default is that the server is not stopped.
Maximum length	3 characters
Format	y, n, yes, no

Parameter	badmessagequeue
Examples	badmessagequeue=SMQ_BADMESSAGE_QUEUE badmessagequeue=bad.message.queue
Description	Specifies the name of the bad message queue. The server puts any MQSeries messages that it does not recognize on this queue.
Maximum length	48 characters
Format	Conforms to the MQSeries queue name specifications.

Inbound ini files

Parameter	inboundqueue
Examples	inboundqueue=yellow.local.queue inboundqueue=SMQ_INBOUND_QUEUE
Description	Specifies the name of the inbound message queue. The inbound server retrieves incoming messages from this queue and converts them to IDocs for input to the receiving R/3 system.
Maximum length	48 characters
Format	Conforms to the MQSeries queue name specifications.

User exit information

These parameters specify whether the named user exit is called by the inbound server.

Parameter	invokeexit
Examples	invokeexit=y invokeexit=n
Description	A flag that specifies whether the named exit is to be used. The default is n.
Maximum length	1 character
Format	y or n

Parameter	exitname
Examples	exitname=d:\saplink\bin\smqesmp2.dll exitname=/usr/ableman/saplinkbin/smqesmp2.a
Description	The name of the user exit DLL or shared library to be used by the exit handler. If you do not specify a path here, you must specify one in the PATH environment variable for your system.
Maximum length	40 characters
Format	Conforms to operating system format for path and file name.

Parameter	exitbuffer
Example	exitbuffer=Luzern
Description	Specifies the optional contents of a buffer passed to the user exit on initialization. You can use this parameter to define a switch to give different user exit functions depending on these contents.
Maximum length	32 characters
Format	Free form text.

Other defaults for the inbound server

The following parameters specify other defaults used by the inbound server:

Parameter	maxconnections
Example	maxconnections=256
Description	Specifies the maximum number of cached handles (recorded connections) from the inbound server to R/3 systems. The default is 256.
Maximum length	4 characters
Format	A number.

Parameter	rfctraceon
Examples	rfctraceon=yes rfctraceon=y rfctraceon=no rfctraceon=n
Description	Option to turn the RFC trace on or off. The trace file is saved in dev_rfc in the current directory. By default, tracing is switched off.
Maximum length	3 characters
Format	y, n, yes, no

Parameter	maxidocsize
Examples	maxidocsize=500,000 maxidocsize=2000
Description	Specifies the maximum size, in bytes, of an inbound IDoc batch. The default is 2,097,152 bytes.
Maximum length	9 characters
Format	numeric

Inbound ini files

Parameter	loadbalance
Examples	loadbalance=yes loadbalance=y loadbalance=no loadbalance=n
Description	Specifies whether the hostname value is to be used as an index into the file <code>saprfc.ini</code>
Maximum length	20 characters
Format	y, n, yes, no

Parameter	logging
Examples	logging=yes logging=y logging=no logging=n
Description	Controls the writing of status messages to the error log.
Maximum length	3 characters
Format	y, n, yes, no

saprfc.ini file information

The values in this file are used by the inbound server if load balancing is to be used.

Parameter	DEST
Examples	DEST=BIN
Description	This should match the value of the <i>hostname</i> defined in the inbound ini file.
Format	alphanumeric

Parameter	TYPE
Examples	TYPE=B
Description	Type of connection (for making a connection to a message server)
Format	alphabetic

Parameter	R3NAME
Examples	R3NAME=BTF (where BTF is the SID)
Description	R/3 system name
Format	alphanumeric

Parameter	MSHOST
Example	MSHOST=9.123.456.7
Description	Host name of message server
Format	IP hostname or IP address

Parameter	GROUP
Examples	GROUP=Z_MQSERIES_LINK
Description	CCMS logon group
Format	alphanumeric

Inbound ini files

Chapter 10. Troubleshooting

This chapter contains information to help you to diagnose and solve problems encountered when using MQSeries link for R/3.

The contents of this section include:

- “Outbound server problem diagnosis”
- “Inbound server problem diagnosis” on page 84
- “Communication and system failures” on page 85
- “Handling unrecognizable messages” on page 86
- “Using trace and error logging to diagnose problems” on page 90

Outbound server problem diagnosis

If an error occurs when the outbound server is processing a message, the following list will help you to diagnose the error:

1. Check that the local queue manager is started.
2. Are the outbound and transaction id queues defined and usable?
3. Do you have access to all queues used by the outbound server?
4. Check that the R/3 RFC destination *gateway host name*, *gateway service*, and *program ID* match the same parameters defined at outbound server initialization.
5. Ensure that the configuration defined in **smqDestConf** has an appropriate entry and has valid queue names.
6. Are the TCP/IP ports defined?
7. Check that sufficient memory has been allocated to handle the size of the processed IDoc batches.
8. If an exit is used, check that the exit has executed successfully.
9. On R/3, check that the ALE configuration is correct.
10. On R/3, verify that the “Partner Profile” configuration (R/3 transaction code we20) has the batch IDoc processing option selected.
11. Ensure that the R/3 database and R/3 instance are active for the outbound R/3 system.
12. On R/3, check that IDocs have not been routed unnecessarily to the R/3 transaction sm58. If R/3 goes down, the outbound server is terminated. When the system is restored, both the outbound and inbound servers must be restarted.

Troubleshooting

You may also find further diagnostic information through the MQSeries home page on the World Wide Web. Refer to “MQSeries information available on the Internet” on page 134 for further information.

Starting the outbound server on OS/390

The following errors can occur when you start the outbound server on OS/390:

Error message

The attempt to connect to Qmgr xxxx failed. Reason code 2129

User action

Check that the environment has the STEPLIB set for the MQSeries adapter. For more information, see the description of setting the STEPLIB on OS/390 on page 40.

Error message

The module librfc was not found

User action

Check that your LIBPATH points to where the librfc module resides.

Inbound server problem diagnosis

On the inbound server, you can get 'invalid IDoc length' errors (SMQ4172) if you do not ensure that each IDoc with an MQSeries message has a unique identifier. See “Constructing messages with IDocs” on page 58 for details.

You may also receive unrecognizable, or bad, messages on the inbound queue. For more information about handling these messages, see “Handling unrecognizable messages” on page 86.

If an error occurs when the inbound server is processing a message, you can use the following list to help diagnose the error:

1. Check that the local queue manager is started.
2. Verify that the inbound, transaction id, and bad message queues are defined and usable.
3. Do you have access to all the queues used by the inbound server?
4. Has enough memory been allocated to handle the size of messages being processed?
5. If an exit is used, verify that the exit completed successfully.
6. Check that the following R/3 connection parameters provide a connection to the required remote R/3 system:
 - Client
 - User
 - Password
 - Language

- System instance number
 - Application server host name
7. Check the bad message queue for messages (refer to “Handling unrecognizable messages” on page 86).
 8. On R/3, check transaction we05 for processed inbound IDocs.

You may also find further diagnostic information through the MQSeries home page on the World Wide Web. Refer to “MQSeries information available on the Internet” on page 134 for further information.

Data Conversion problems

If you receive conversion warnings such as RC2119 on MQGET, or the inbound server reports that the inbound message header is invalid, check the following:

- That the data conversion exit is in the path and can be found by MQSeries. On an MQSeries Version 5 server this path is in the **QM.INI** file.
- That the CCSIDs and encoding are correct for the queue managers.
- That the message format string in the message descriptor of the MQSeries message is set to the appropriate value. If a user exit is not called this should be set to 'MQHSAP'.
- That you have the latest service applied if you are using an MQSeries Version 5 server.

Communication and system failures

Communication between MQSeries link for R/3 and R/3 uses TCP/IP network protocol services and therefore any failure with the TCP/IP network should be referred to your network administrator. However, with the assistance of the checklists given earlier in this chapter, the user can ensure that the problem is not connected with the MQSeries link for R/3 configuration before passing the problem to the TCP/IP administrator.

MQSeries link for R/3 can detect problems with connection to the R/3 database layer or the R/3 application server layer. If these layers become unavailable, the SAPGUI will display an error message. Inform your SAP systems administrator of the problem and do not continue to use the R/3 system.

What happens when MQSeries link for R/3 detects a failure

Depending on what stage of MQSeries link for R/3 processing the systems failure occurs, the product will handle the transmitted message as follows:

- On outbound processing:
 1. Rolls back the unit of work (UOW)
 2. Logs the error

Troubleshooting

3. Reports the error to R/3, where it is also logged
 4. Moves the IDoc message to the Transactional RFC panel (R/3 transaction code sm58), where it can be retried after you have corrected the error
- On inbound processing:
 1. Rolls back the unit of work (UOW)
 2. Logs the error
 3. Moves the IDoc message back to the inbound queue or to the bad message queue, depending on the type of failure that has occurred

For more information on error logging, refer to “Using trace and error logging to diagnose problems” on page 90.

Handling unrecognizable messages

Unless a data conversion exit is to be used, when the inbound server is waiting for a message to appear on the inbound queue it is looking for the information in a recognizable format. There must be an MQSeries link for R/3 header at the front of the message and the information contained in the message must be recognizable to R/3. In other words, the information must be in IDoc format (for more information about the prerequisites for incoming messages see “Message formats” on page 55. If a user exit is to be used, the inbound message must meet the requirements of the user exit.

But what happens when a message appears on the queue that the inbound server cannot recognize? It would be inappropriate to leave the ‘bad message’ on the inbound queue because the inbound server would continually try to retrieve the message and fail (while the queue is set to its default, FIFO), rendering the inbound queue unusable to the inbound server. The inbound server solves this problem by using a *bad message queue* as a repository for these unrecognizable messages, thus clearing the inbound queue to enable processing of subsequent messages.

The inbound server provides a degree of flexibility for the handling of bad messages. An optional parameter in the inbound ini file, `terminateonbadmessage`, controls whether or not the inbound server terminates when an unrecognizable message is encountered. By default, the inbound server will terminate when a bad message is encountered. If this parameter is set to no, the server will place the message on the bad message queue, the name of which is also defined as a mandatory parameter in the inbound ini file.

Contents of a bad message

Bad messages are placed onto the bad message queue with a bad message header appended as shown in Figure 13.

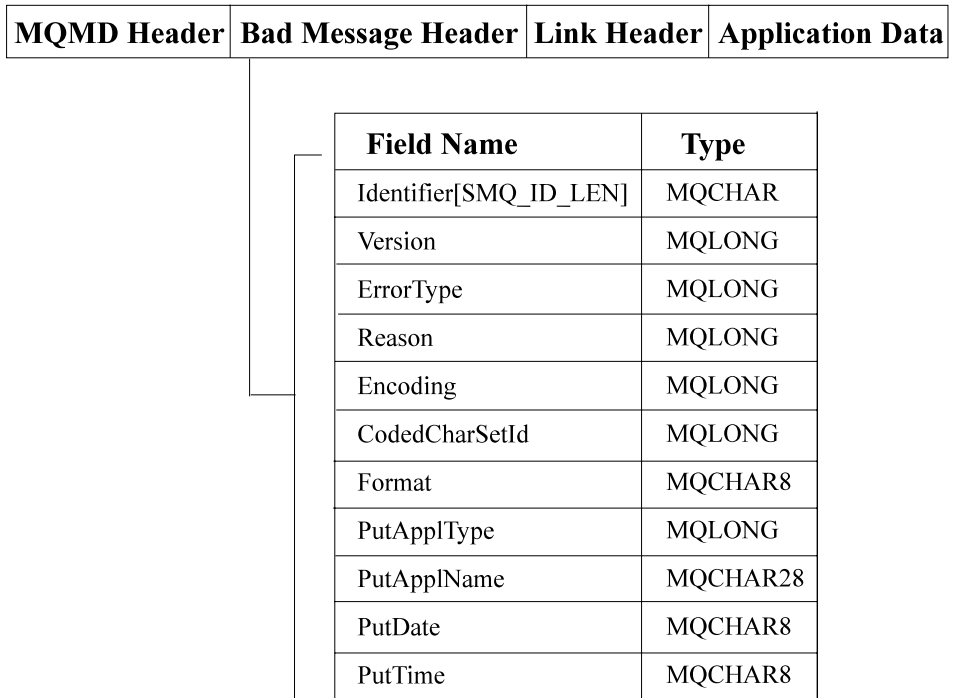


Figure 13. Structure of the bad message header.

Note: The information included in the bad message header is not intended for bad message diagnosis. To perform this task bad message error codes and bad message error types are included in MQSeries link for R/3. Furthermore, the majority of information included in the bad message header appears in hexadecimal format, when viewed with a standard text editor, thus making it difficult to perform any analysis on the header using basic editor tools. Further information can be found about the structure of the bad message header in the supplied `smqc.h` include file.

Bad message diagnosis

When a bad message is generated, the user receives on-screen confirmation that the message has been placed on the bad message queue. Also displayed is the reason code for its classification as a bad message, and the error type.

Note: If the server is started automatically, *standard error* and *standard output* should be captured to a file so that the bad message information can be

Troubleshooting

viewed later. There are two possible error types. Their explanations, and the actions you should take to resolve them are shown in Table 1.

Table 1. Error types

Error type	Explanation	Action
1	Either an error was discovered in the MQSeries link for R/3 header, or the IDoc data in the message was invalid.	Look up the bad message reason code in Table 2 and follow the course of action.
2	An error has occurred within the MQSeries processing of the message.	Lookup the reason code in the MQSeries Application Programming Reference manual and follow the course of action specified.

Bad message reason codes

Notes:

1. `b` is a blank character (ASCII 20_x, EBCDIC 40_x).
2. `<data>` represents the string returned in the explanation message.

Table 2. Reason codes

Reason code	Explanation	Action
4102	The user exit returned SMQRC_CONVERT_FAIL.	Check the user exit to determine why the conversion failed.
4104	The user exit returned SMQRC_CONVERT_BAD_MESSAGE.	Check the user exit to determine why the message is bad.
4105	The user exit returned SMQRC_TERMINATE_EXIT.	The user exit is terminating. Check the user exit to determine why the conversion failed.
4106	The user exit returned SMQRC_TERMINATE_SERVER.	The server is terminating at the request of the user exit. Check the exit to determine why it requests the server to be terminated.
4107	IDoc has an invalid structure header. IDoc value= <code><data></code> . Expected value was "SMQ b ".	Amend the MQSeries link for R/3 header identifier, to "SMQ b ".

Table 2. Reason codes (continued)

Reason code	Explanation	Action
4108	IDoc has an invalid structure version. IDoc value=<data>. Expected value was “bbb1”. Must be numbers.	Amend the MQSeries link for R/3 header version number to “bbb1”.
4109	The length of the inbound message is invalid for an IDoc.	Amend IDoc data.
4110	IDoc has an invalid language. IDoc value=“<data>”	Amend the MQSeries link for R/3 header language (E for English).
4111	IDoc has an invalid client. IDoc value=“<data>”	Amend the MQSeries link for R/3 header client number.
4112	IDoc has an invalid system number. IDoc value=“<data>”	Amend the MQSeries link for R/3 header system instance number.
4113	RfcOpen failed. Failed to connect to SAP system for inbound IDoc. Check host and system names.	Check that the R/3 system can be contacted and the system instance number and the host name of the remote R/3 system are valid. Amend this information in the MQSeries link for R/3 header if necessary.
4114	Failed to get SAP transaction Id. Check Client, User ID, Password, and Language.	Check the remote R/3 connection parameters and the availability of the source and destination R/3 systems.
4115	The attempt to put the message into SAP failed.	Check the format of the IDoc and the ALE partner profile.

Reprocessing bad messages

To allow a bad message to be reprocessed by the inbound server, follow this procedure:

1. Using the reason code given for bad message generation, amend the message according to the instructions in the corresponding 'Action' column.
2. Remove the bad message header from the message.
3. Place the amended bad message back onto the inbound queue. The message will now be processed to the remote R/3 system.

Troubleshooting

Note: The administrator has two options. The process can either be performed manually, or a program can be written that utilizes information in the bad message header to automatically handle any bad messages.

Using trace and error logging to diagnose problems

There are two types of trace available with MQSeries link for R/3 that assist you to undertake a step by step analysis of a problem.

MQSeries link for R/3 trace

MQSeries link for R/3 trace is provided primarily for the use of IBM service personnel. It traces the functions executed by MQSeries link for R/3. A trace file is generated in the current directory each time a server is started, the file being named SMQnnnnn.TRC, where nnnnn is a unique, non-sequential, five-digit number. You activate this trace by adding SMQ_TRACE=F and SMQ_TRACE_LEVEL=2 to the list of *user* environment variables on the system where the servers are being run.

If you are running the outbound or inbound servers as Windows NT services, the SMQ_XLAT_PATH=d:\smq\bin, SMQ_TRACE=F, and SMQ_TRACE_LEVEL=2 statements must be present in the *system* environment variables. You can use the following procedures to add these statements:

1. Open the System settings window by following the path, Start\Settings\ControlPanel\System.
2. Add the entries to the *system* list as follows:

Variable	Value
SMQ_XLAT_PATH	d:\smq\bin
SMQ_TRACE	F
SMQ_TRACE_LEVEL	2

Contact your systems administrator for more information on system environment variables.

A TRC trace file is created for each instance of an inbound or outbound server.

MQSeries link for R/3 Error logging

As with other MQSeries family products, MQSeries link for R/3 generates three types of messages, which, in order of increasing severity are:

- Information messages
- Warning messages
- Error messages

For a complete list of messages, see “Appendix C. Messages and codes” on page 109 .

All errors are logged in a set of log files: SMQERR01.LOG, SMQERR02.LOG, and SMQERR03.LOG, which are written to in this order. When one file is filled, the next is started. When all three are filled, the log starts again at SMQERR01.LOG.

When the MQSeries link for R/3 servers are started, a log file repository is created as the errors sub-directory. All log files are placed in this repository. The sub-directory resides under the current active directory.

If the servers are running as a WINNT service the trace is stored in the **system** directory and the error logs are placed in the **system\errors** subdirectory.

You can control the writing of status messages to the error log by setting the *logging* parameter in the INI file:

logging=y|n

This parameter controls whether IDoc status logging to the SMQERRxx.LOG files is enabled or disabled.

IDoc status logging to the console is unchanged and will still appear as before. It is important to understand that *all* error and warning messages are still logged to the error log files. The logging option does not affect the logging of errors and warnings.

You set this parameter in the in.ini file for the inbound server and in the out.ini file for the outbound server. See “Sample initialization files” on page 95 in “Appendix A. Samples” on page 95.

Note: To use the *logging* parameter, you must have installed CSD01 for MQSeries link for R/3.

Error logging on OS/390

On OS/390, log messages are written to screen and to file. Some messages are also written to the operator console and to SYSLOG.

Note: If your MQSeries system is heavily used, a large number of information messages are sent to the OS/390 console and hardcopy log. You can suppress some of these console messages by using the OS/390 message processing facility list, specified by the MPFLSTxx members of SYS1.PARMLIB. Refer to the *MVS Initialization and Tuning Reference* book for more information about MPFLSTxx.

Trace

R/3 Error Logging

If R/3 encounters an error while trying to send an IDoc outbound from R/3, the error message is reported back to R/3 and the UOW is rolled back. This roll back occurs whether it is a systems error, an MQSeries error, or an MQSeries link for R/3 configuration error.

The error is logged by SAP under R/3 transaction code sm58. The administrator can view the error text associated with the message and take the appropriate action to address the error. The message can also be retrieved from R/3 transaction code sm58.

R/3 Remote function call (RFC) trace

When you initialize the server, you can specify whether an RFC trace is to be generated. The trace file is generated by setting `rfctraceon=y` at outbound or inbound initialization. Once the file is created, any further instances of outbound or inbound errors will append trace information to the same file.

Note: On the Solaris platform there is a bug in some versions of the RFC library tracing function and this causes a core dump. The problem is fixed in versions 3.1G or later of R/3.

Chapter 11. Security

You can use the security features of MQSeries and MQSeries link for R/3 to ensure that only authorized users have access to MQSeries message queues. MQSeries link for R/3 maintains all the security features provided by R/3. MQSeries link for R/3 can put your IDocs into R/3, but only if you have an authorized user ID and password for that system.

When you start an MQSeries link for R/3 server, the server application must be able to connect to the specified queue manager and open the required queues for gets or puts (reading or writing to queues), depending on the type of server and the type of queue.

Startup security

If the MQSeries Object Authority Manager is turned on, any server on that workstation must run under a user ID that is a member of the group under which MQSeries is installed. This is usually 'mqm'. Otherwise, the server program cannot connect to the queue manager and the server cannot start.

Security precautions

To ensure the integrity of SAP passwords, you should ensure that you do not specify SAP passwords in the **smqDestConf** file. If you do, the outbound server builds the password into the link header of each message it generates.

To prevent passwords being transmitted over the network:

1. Specify the password of the destination R3 system in the inbound ini file, using the *password* parameter.
2. Use the operating system to grant access to any MQSeries or MQSeries link for R/3 files only to trusted users. In particular, restrict access to the inbound **ini** file and any files associated with the inbound message queue.

Alternatively, you could consider encrypting the passwords at the outbound server using user exits, and decrypting them at the inbound server.

You can also write additional security checks, using the security user exits supplied by MQSeries.

Appendix A. Samples

This appendix contains information about the sample files supplied with MQSeries link for R/3. The samples include:

- Initialization (ini) files
- Configuration file
- User exits
- MQSC command files

Sample initialization files

File Name	Purpose
out.ini	Sample initialization file for use with the outbound server. This file contains a set of optional and required parameters. You can use this file as a basis for your own ini file. For more information about outbound server ini file parameters, see “Outbound server initialization parameters” on page 71. At installation time this file is installed in the samp directory.

Note: On OS/400 the out.ini sample initialization file is located in directory /QIBM/ProdData/smq/samp.

```
gatewayhost=ha0016
gatewayservice=sapgw86
programid=telstaret
exitname=d:\saplink\bin\smqesmp1.dll
exitbuffer=LosAndos
rfctraceon=n
transactionqueue=orange.local.queue
queuemanager=saturn.queue.manger
maxidocsize=200000
batching=y
logging=y
```

Figure 14. Sample outbound ini file showing sample parameters

in.ini	Sample initialization file for use with the inbound server. This file contains a set of optional and required parameters. You can use this file as a basis for your own ini file. For more information about inbound server ini file parameters, see “Inbound server initialization parameters” on page 75. This file is located in the samp directory.
---------------	---

Samples

Note: On OS/400 the **in.ini** sample initialization file is located in directory **/QIBM/ProdData/smq/samp**.

```
client=100
user=hursley5
language=e
password=leisa
sysnbr=00
hostname=BIN
transactionqueue=SMQ_INBOUND_TRANIDS
queuemanager=ehningen.qmgr
terminateonbadmessage=n
badmessagequeue=SMQ_BADMESSAGE_QUEUE
inboundqueue=SMQ_INBOUND_QUEUE
invokeexit=n
exitname=d:\saplink\bin\smqesmp1.dll
exitbuffer=Abe1Seaman
maxconnections=256
rfctraceon=n
maxidocsize=05000000
loadbalance=y
logging=y
```

Figure 15. Sample inbound ini file showing sample parameters

saprfc.ini Sample **saprfc.ini** file that enables load balancing support for the inbound server. For more information about this file see “saprfc.ini file information” on page 80.

```
DEST=BIN
TYPE=B
R3NAME=BTF
MSHOST=9.165.255.88
GROUP=Z_MQSERIES_LINK
```

Figure 16. sample rfc.ini file

Note: In Figure 15 and Figure 16 the *hostname* and *DEST* parameters are equated to the same value (BIN) as required for load balancing. (See “Inbound load balancing” on page 7.)

Sample configuration file

The **smqDestConf** is a sample configuration file that contains the information to map R/3 logical destinations to MQSeries destinations. The file contains a series of blocks of information each of which maps an IDoc type to an MQSeries destination. See “Task 5. Mapping R/3 Logical Systems to MQSeries destinations” on page 35 for more information about the parameters that can be set in this file.

The sample file is in the **samp** directory, but the edited version for use with your system should be stored in the directory that the outbound server is run from.

Samples

```
#####  
**                                                                 **  
** Statement:      Licensed Materials - Property of IBM          **  
**                                                                 **  
**              04L1641, 5765-B66                                **  
**              04L1661, 5765-C01                                **  
**              04L1671, 5765-B98                                **  
**              04L1681, 5765-B99                                **  
**                                                                 **  
**              (C) Copyright IBM Corp. 1996, 1998              **  
**                                                                 **  
#####  
#####  
** Module Name: smqDestConf                                       **  
** Type          : sample destination configuration file for the MQSeries **  
**              : Link for R/3.                                     **  
** Function      : Map R/3 logical destinations to MQSeries destinations. **  
**              :                                                                 **  
#####  
#####  
** The first destination. It is also the default destination.    **  
** The default destination is marked by setting the default      **  
** parameter to yes. It does not have to be the first destination. **  
** This destination is sending information to another R/3 system  **  
#####  
receivingdestination  
receivingpartner=ORDERS0  
edi_mestype=MATMAS  
outboundqueuemanager=ORDERS.QMGR  
outboundqueue=SMQ_OUTBOUND_QUEUE  
calluserexit=no  
exitname=  
exitbuffer=  
client=100  
language=e  
hostname=machine.location.company.com  
systemnumber=00  
userid=user1  
password=password1  
default=yes
```

Figure 17. Sample smqDestConf file (Part 1 of 2)

```

#*****#
#* The second destination.                                *#
#* This destination is not sending to another R/3 system. *#
#*****#
receivingdestination
receivingpartner=ORDERS1
edi_mestype=MATMAS
outboundqueueanager=ORDERS.QMGR
outboundqueue=ORDERS1.QUEUE
calluserexit=yes
exitname=/var/mqm/smq/exts/EXIT1
exitbuffer=
client=
language=
hostname=
systemnumber=
userid=
password=

#*****#
#* The third destination.                                *#
#*****#
receivingdestination
receivingpartner=ORDERS2
edi_mestype=*
outboundqueueanager=ORDERS.QMGR
outboundqueue=ORDERS2.QUEUE
calluserexit=no
exitname=*
exitbuffer=
client=
language=
hostname=
systemnumber=
userid=
password=

```

Figure 17. Sample smqDestConf file (Part 2 of 2)

Sample user exits

The two sample C source files described below are supplied with MQSeries link for R/3.

File Name	Purpose
smqesmp1.c	This sample shows the structure of a user exit. You can use it in either an outbound or an inbound server. For an outbound server the exit gets a buffer large enough to hold both the header and the message data passed to it. It copies the header and the data to this buffer. This is then passed back to the exit handler in the output structure with a Result field set to CONVERT_OK.

Samples

For an inbound server, the exit expects the message data passed to it in this format:

```
<MQSAPH> <IDoc control> <IDoc data> and so on.
```

The sample allocates a buffer large enough to hold this data and passes this back to the exit handler. The data returned has separate pointers to the header and to the IDoc data. The Result field is set to CONVERT_OK. If the input data is not in the format expected, the exit returns with a result of CONVERT_NOT_NEEDED.

smqesmp2.c

This sample shows the transformation of:

- An inbound message that is not in the R/3 IDoc format to the IDoc format required by the R/3 Materials Master function.
- An outbound message in IDoc format (as produced by the R/3 Materials Master function) to a user-defined format suitable for processing by a non-SAP system. This is the reverse of the inbound transformation. The user-defined formats are defined in the sample header file, **smqeshd1.h**.

This sample also shows the use of MQI calls within an exit.

See “Chapter 6. Writing user exits” on page 45 for more information about how to use the exits.

The header files associated with these source files are also supplied in the same samples directory.

Note: The source file supplied with Version 1.2 works with Version 3 IDocs only. However, the MQSeries link for R/3 product works with Version 3 and Version 4 IDocs.

Sample data conversion exit

This section describes the sample data conversion exits supplied with MQSeries link for R/3. The information in this section applies only if you are running Version 5.0 or an earlier version of MQSeries.

Version 5.1 and later versions of MQSeries support MQHSAP message header conversion, so the MQHSAP exit from earlier versions of MQSeries (located in the directory `/var/mqm/exit`) is not required with MQSeries Version 5.1 and later versions. If you have such an exit that you used with an earlier version of MQSeries then you should delete it to avoid potential user data conversion problems.

If you use your own program to put messages to the inbound queue when using MQSeries link for R/3 with MQSeries Version 5.1 or later, you should set the *Format* field in the MQHSAP header to MQSTR to describe the IDoc data that follows the MQHSAP header.

Sample exit for distributed platforms

Note: This information applies only when you are running MQSeries link for R/3 on AIX, Compaq Tru64 UNIX, HP-UX, Sun Solaris, OS/400, and Windows NT.

MQSeries link for R/3 supplies an MQSeries Data Conversion exit that is used when MQSeries passes data between systems that require data conversion. More information about data conversion can be found in *MQSeries Application Programming Guide*, SC33-0807 and *MQSeries Application Programming Reference*, SC33-1673 books.

smqsdcnv.c is the sample data conversion exit supplied with MQSeries link for R/3. It is also supplied as a non-threaded executable MQHSAP.

MQSeries automatically calls this exit if data conversion is required. The exit treats all data after the MQSeries link for R/3 link Header as character data.

Notes:

1. On Compaq Tru64 UNIX, the data conversion exit library MQHSAP is not required because data conversion for the sample data conversion exit supplied with MQSeries link for R/3 header can now be done in MQSeries for Compaq Tru64 UNIX (but is shipped here for backward compatibility).
2. On OS/400 the sample user exit source is in file **C** in library **QMQLINK**.

Sample data conversion exit for OS/390

On OS/390 you must have installed MQSeries Server for OS/390 Version 5.2 with the PTF applied to fix APAR PQ44683. This version supports the MQHSAP header.

If your installation is running an earlier version of MQSeries that might receive messages containing an MQHSAP header, you can use the sample data conversion exit that is supplied with MQSeries link for R/3 on OS/390.

MQSC command file (smqscdef.tst)

The MQSC command file **smqscdef.tst** is located in the **samp** directory.

Note: On OS/400 the sample MQSeries command source is in file QMQSC in library QMQLINK.

See “Understanding which MQSeries objects you need” on page 23 for more information on the use of this file.

The file contains the following definitions:

SMQ_INBOUND_TRANIDS¹

Inbound transaction ID queue

This queue stores R/3 transaction IDs for inbound messages.

SMQ_OUTBOUND_TRANIDS¹

Outbound transaction ID queue

This queue stores R/3 transaction IDs for outbound messages.

SMQ_OUTBOUND_QUEUE

Outbound message queue

The outbound server uses this queue to store outbound messages where they can be retrieved by another application. The server builds these messages from IDocs flowing from an R/3 system. The queue name must match the one specified in the **smqDestConf** file..

SMQ_INBOUND_QUEUE

Inbound message queue

The inbound server uses this queue to get inbound messages from MQSeries. The server processes each message, decomposing it into IDocs, which it then passes to the destination R/3 system. The queue name matches the inbound queue name specified in the sample inbound ini file.

SMQ_BADMESSAGE_QUEUE

Bad message queue

The inbound server uses this queue to store any messages that it is unable to convert into valid IDocs. You must ensure that the name of this queue matches the queue name specified in the sample inbound ini file.

Notes:

1. SMQ_OUTBOUND_TRANIDS and SMQ_INBOUND_TRANIDS are the default names for the transaction ID store queues. If you use these names, you can run MQSeries link for R/3 without explicitly specifying the transaction queue names in the outbound and inbound ini files.

2. All queues are defined with the default persistence (DEFPSIST) attribute set. All MQSeries messages generated by the link are persistent messages.

OS/390 samples

The following sample JCL is supplied as members of the thlqual.SASQJCL partitioned data set, where thlqual is the target high level qualifier of the MVS data set for MQSeries link for R/3:

ASQRUN

This file contains an example of starting the outbound server. You must modify this file to suit your environment.

ASQSTOP

This file contains an example of stopping the outbound server. It requires smqstop to be available in the /usr/lpp/smq/samp directory on the HFS.

ASQQUERY

This file contains an example of how you can query the running servers.

ASQMJOB

This file contains JCL to configure the sample MQ objects needed by MQSeries link for R/3.

ASQCMLS

This file contains the MQSeries object definitions that are used as input for ASQMJOB.

ASQSAPHX

This file contains sample JCL to build the MQHSAP data conversion exit for use with MQSeries Version 2.1, because this version of MQSeries does not support the MQHSAP header.

The following sample source is supplied as a member of the thlqual.SASQASM partitioned data set, where thlqual is the target high level qualifier of the MVS data set for MQSeries link for R/3:

MQHSAP

This file contains the assembler program source for the data conversion exit.

Samples

Appendix B. Quick reference

This section summarizes the configuration parameters for the outbound and inbound servers. You can use these tables to check your own configurations. Table 4 on page 106 summarizes those for inbound configuration; Table 3 summarizes the outbound configuration parameters,

Outbound configuration

Table 3. Outbound configuration parameters

smqDestConf file parameters ¹	Command line parameters on smqso ¹	Outbound initialization file ¹
-	-iInitializationFileName ²	-
Queue manager name ^{3a}	-	-
-	-mQueueManagerName ^{3b}	queuemanager ^{3b}
Queue name ⁴	-	-
User Exit	-	exitname ⁵
Exit Buffer ⁶	-	exitbuffer ⁶
Call user exit ⁷	-	-
-	-	rfctraceon
-	-	transactionqueue
-	-	maxidocsize
-	-	batching
	-xGatewayService ⁸ -gGatewayHost ⁸ -aProgramID ⁹	gatewayservice ⁸ gatewayhost ⁸ programid ⁹

Notes:

1. The descending precedence of parameters is: command line, **ini** file, **smqDestConf** file.
2. This parameter is optional for the outbound server.
3. We recommend that the same queue manager is used for both these definitions although technically they are not necessarily the same:
 - a. The queue manager specified in the **smqDestConf** file is the queue manager that owns the outbound queue.

Quick reference

- b. The queue manager specified on the command line or in the **ini** file specifies the queue manager that owns the transaction ID store queue.

If a queue manager name is not specified, the default queue manager is used.

4. Queue name is a required parameter and must be defined in the **smqDestConf** file.
5. For the server to invoke this user exit name, you must also specify an asterisk (*) in the *userexit* field and set *calluserexit* to 'yes' in the **smqDestConf** file.
6. The use of an exit buffer is optional.
7. See note 5.
8. Gateway service and gateway host, if specified in the RFC destination panel, must match the values actually used on the smqso command (taken from the **ini** file or command line parameters).
9. You define the program ID. The value you specify in the RFC destination panel must match the values actually used on the command (taken from either the **ini** file or from the command line parameters). Note that the program ID is entered as a character string that matches the name specified in the R/3 RFC destination panel.

Inbound configuration

Table 4. Inbound configuration parameters

Command line parameters on smqsi ¹	Inbound initialization file ¹
-iInitializationFileName ²	-
-	client ³
-	language ³
-	userid ³
-	password ³
-	hostname ³
-	sysnbr ³
-	-
-	transactionqueue
-mQueueManagerName	queuemanager
-	terminateonbadmessage
-	badmessagequeue
-qInboundQueueName	inboundqueue
-	exitname

Table 4. Inbound configuration parameters (continued)

Command line parameters on smqsi ¹	Inbound initialization file ¹
-	exitbuffer
-	maxconnections
-	maxidocsize
-	rfctraceon
-	loadbalance

Notes:

1. The descending precedence of parameters is: command line, **ini** file
2. This parameter is required for the inbound server.
3. These parameters can also be passed in the message link header if they are defined in the smqDestConf on the outbound server.

Quick reference

Appendix C. Messages and codes

The messages and codes shown in this appendix are generated by MQSeries link for R/3.

Note: Some messages have double asterisks (**) surrounding them; these messages appear on the panel associated with R/3 transaction code sm58 and the ** is to draw attention to them. They are described in this appendix.

SMQ4101.MSG

The MQSeries Link for R/3 server has started.

Explanation: The MQSeries Link for R/3 server started normally.

Action: None.

SMQ4102 **The MQSeries Link for R/3 server has ended normally.**

Explanation: The MQSeries Link for R/3 server completed normally.

Action: None.

SMQ4103 **The MQSeries Link for R/3 server failed to start. See other messages for details.**

Explanation: The MQSeries Link for R/3 server failed to start. See other messages for a more detailed explanation.

Action: Check other error messages, correct the errors and re-run the server.

SMQ4104 **The MQSeries Link for R/3 server ended abnormally. See other messages for details.**

Explanation: The MQSeries Link for R/3 server ended abnormally. See previous messages for a more detailed explanation.

Action: Check error messages, correct the errors and re-run the server. If the problem persists, note the errors and reason code and contact your IBM representative.

SMQ4105 **Could not start trace services. Return code <return code>**

Explanation: The MQSeries Link for R/3 server could not start trace services. Return code <return code>.

Action: Check that the trace environment variables have been correctly specified.

SMQ4106 Server failed to open the Destination Configuration File.

Explanation: Either the file could not be found or the Outbound Server does not have sufficient access permission.

Action: Resolve the problem and re-run the server.

SMQ4107 The ini file - <file name> - could not be found.

Explanation: The name of the ini file specified on the command line was not valid or the file does not exist.

Action: Check that the file name and path are correct.

SMQ4108 Invalid ini file parameter at line <line number>.

Explanation: An error was detected on line <line number> of the ini file.

Action: Correct the error and run the server again.

SMQ4109 The attempt to connect to Qmgr <queue manager name> failed. Reason code <reason code>.

Explanation: The MQCONN call to queue manager <queue manager name> failed. The reason code from the MQCONN call was <reason code>.

Action: Investigate the MQSeries error code, fix the error and re-run the server.

SMQ4110 The exit handler detected an invalid state.

Explanation: An internal error occurred while processing a user exit.

Action: Please contact your IBM representative.

SMQ4111 The exit handler has not been initialized.

Explanation: An internal error occurred while processing a user exit because an exit handler component was called before the exit handler was initialized.

Action: Please contact your IBM representative.

SMQ4112 Invalid instance handle passed to exit handler.

Explanation: An internal error occurred while processing a user exit.

Action: Please contact your IBM representative.

SMQ4113 Invalid entry handle passed to exit handler.

Explanation: An internal error occurred while processing a user exit.

Action: Please contact your IBM representative.

SMQ4114 The maximum number of exit instances has been exceeded.

Explanation: A maximum of 256 exits can be run at one time by the exit handler, and this number has been exceeded.

Action: Reduce the number of exits that are run at one time.

SMQ4115 The maximum number of <number> exit entries has been exceeded.

Explanation: A maximum of <number> exits can be run at one time by the exit handler, and this number has been exceeded.

Action: Reduce the number of exits that are run at one time.

SMQ4116 Exit <exit name> has already been loaded in inbound.

Explanation: An internal error occurred while processing a user exit.

Action: This is an information message. No action required.

SMQ4117 Exit <exit name> has already been loaded in outbound.

Explanation: An exit with the same name has already been loaded in outbound and will therefore not need initializing again.

Action: This is an information message. No action required.

SMQ4118 Invalid command line arguments. See extended text or documentation.

Explanation: Syntax for start of SAP -> MQSeries outbound server : smqso [options] with options =

-a<Program ID> e.g. <own host name>.mqserver
-g<SAP gateway host name> e.g. hs0311
-x<SAP gateway service> e.g. sapgw01
-m<local queue manager name> e.g. qmgrtest

(the queue manager name is optional if you have installed and wish to use a default queue manager)

Action: Correct the arguments and re-run the server.

SMQ4119 Syntax error at line <line number> of ini file.

Explanation: An error was detected on line <line number> of the ini file.

Action: Modify the ini file parameter at line <line number> and restart the server.

SMQ4120 Missing '=' at line <line number> of ini file.

Explanation: The lines in the ini file must have the format: parameter=value but an equal sign is missing from a parameter at line <line number>.

Action: Correct the error and run the server again.

SMQ4121 **The connect to R/3 failed. Check that the R/3 system is available and the gateway host, service and program ID are correct.**

Explanation: The attempt to connect to the SAP system failed. Check that the R/3 system is available, and that the gateway service, hostname, and program ID are correct. If the error can not be found, enable RFC trace using the option in the ini file, re-run the server and look at the the dev_rfc trace file.

Action: Correct the errors and re-run the server.

SMQ4122 **The attempt to open the default queue manager failed with reason code <reason code>.**

Explanation: The MQOPEN attempt to open the default queue manager failed with MQSeries reason code <reason code>.

Action: Check the MQSeries reason code <reason code>, correct the error and re-run the server, or specify the name of the queue manager to use.

SMQ4123

Outbound MQSeries link for R/3 - service level 1.20 , Date:

Syntax for outbound server:

smqso [options]

with options=

-a<Program ID> e.g. <own host name>.mqserver
 -g<SAP gateway host name> e.g. hs0311
-x<SAP gateway service> e.g. sapgw00
 -m<local queue mgr name> e.g. qmgrtest
 -i<initialisation file> e.g. out.ini

(The queue manager name is optional if you have
(installed, and wish to use, a default queue manager.)

Explanation: The syntax of your command was incorrect.

Action: Change the values specified on the command line and re-run the command.

SMQ4124

Inbound MQSeries link for R/3 - service level 1.20 , Date:

Syntax for inbound server:

smqsi [options]

with options

-i<initialisation file> e.g. in.ini (required)
 -q<local queue> e.g. inbound.idocs
 -m<queue mgr> e.g. qmgrtest

(The queue manager and queue names are optional and
(will override the ini file values if specified.)

Explanation: The syntax of your command was incorrect.

Action: Change the values specified on the command line and re-run the command.

SMQ4125 **The attempt to open the transaction queue failed with reason code <reason code>.**

Explanation: The MQOPEN attempt on the transaction queue <queue name> failed with MQSeries reason code <reason code>.

Action: Check MQSeries reason code <reason code>, correct the error and re-run the server.

SMQ4126 **The MQSeries Link for R/3 server ended abnormally. See FFST for details.**

Explanation: The MQSeries Link for R/3 server ended abnormally - Reason code <reason code>. See FFST for a more detailed explanation.

Action: Please contact your IBM representative.

SMQ4127 **Failed to execute exit <exit name>.**

Explanation: The MQSeries Link for R/3 server detected an error when attempting to call the exit's execute or return function.

Action: Check that the exit exists in the path specified. Ensure that the exit is not terminating unexpectedly.

SMQ4128 **The MQPUT to queue <queue name> failed with reason code <reason code>.**

Explanation: The attempt to put a message to the outbound queue <queue name> failed with MQSeries reason code <reason code>.

Action: Fix the error and re-run the server.

SMQ4129 **The MQPUT to queue <queue name> failed with reason code <reason code>.**

Explanation: The attempt to put a message to the transaction queue <queue name> failed with MQSeries reason code <reason code>.

Action: Fix the error and re-run the server.

SMQ4130 **The MQGET from queue <queue name> failed with reason code <reason code>.**

Explanation: The attempt to get a message from the transaction queue <queue name> failed with MQSeries reason code <reason code>.

Action: Fix the error and re-run the server.

SMQ4131 **The MQCMIT call failed with reason code <reason code>.**

Explanation: The attempt to commit a transaction failed with MQSeries reason code <reason code>.

Action: Fix the error and re-run the server.

SMQ4132 **The MQBACK call failed with reason code <reason code>.**

Explanation: The attempt to back out a transaction failed with MQSeries reason code <reason code>.

Action: Fix the error and re-run the server.

SMQ4133 **Error. <number 1> IDocs were expected but only <number 2> were received.**

Explanation: SAP reported that <number 1> IDocs were to be sent, but only <number 2> were received.

Action: Try resending the IDocs from SAP. If the problem persists, contact your IBM representative.

SMQ4134 **A SAP RFC communication error occurred. Turn on RFC trace for details.**

Explanation: A SAP RFC communication error occurred.

Action: Use the ini file option to turn on RFC trace, rerun the server, and check the dev_rfc trace file for details.

SMQ4135 **The memory allocation for the exit anchor block failed.**

Explanation: Not enough memory could be allocated for the exit handler.

Action: Try to free up some system resources and run the server again.

SMQ4136 **An error occurred while attempting to load the user exit**

Explanation: The server failed to load the user exit. The error code from the operating system 'load' function was <number 1>.

Action: Check that the exit exists.

SMQ4137 **An error occurred while attempting to unload the user exit**

Explanation: The server could not unload the user exit.

Action: If the error persists, re-run the server with trace enabled and contact your IBM representative.

SMQ4138 **The exit handler has run out of handle space**

Explanation: The exit handler has attempted to load more exits than are allowed.

Action: Re-run the server. If the problem persists please contact your IBM representative.

SMQ4139 **Invalid module handle in exit handler.**

Explanation: An internal error occurred while processing a user exit.

Action: Please contact your IBM representative.

SMQ4140 Failed to get pointer to exit anchor block

Explanation: An internal error occurred while processing a user exit.

Action: Please contact your IBM representative.

SMQ4141 Failed to get entry point to exit

Explanation: The server could not call the user exit because the exit entry point ('Initialize') was not found.

Action: Check that the exit has been written and compiled correctly.

SMQ4142 ** Failed to get tables for IDocs **

Explanation: An error occurred when attempting to get data from the SAP system.

Action: Re-run the server with RFC trace enabled. Look at the trace file and attempt to correct the error.

SMQ4143 More than one default destination has been defined.

Explanation: Only one default destination should be defined in the destination definition file.

Action: Review the definitions in the destination definition file, and make sure there is only one default destination defined.

SMQ4144 No default destination has been defined.

Explanation: One default destination should be defined in the destination definition file.

Action: Review the definitions in the destination definition file, and make sure there is only one default destination defined.

SMQ4145 ** Failed to initialize exit handler **

Explanation: An internal error occurred while processing a user exit.

Action: Please contact your IBM representative.

SMQ4146 ** Failed to copy IDocs into memory buffer **

Explanation: An error occurred when attempting to write data into memory, check other error messages for more details.

Action: Fix the error and resend the IDocs.

SMQ4147 ** The RfcSendData failed **

Explanation: An error occurred when attempting to send data to SAP.

Action: Re-run the server with RFC trace enabled for more information.

SMQ4148 ** User exit <exit name> failed. **

Explanation: An error occurred while processing a user exit.

Action: See other error messages for more information.

SMQ4149 ** MQPUT failed - Reason <reason code> **

Explanation: The MQPUT failed with MQSeries reason code <reason code>.

Action: Please fix the error and send the IDoc again.

SMQ4150 ** Failed to read IDocs from ITABs **

Explanation: An error occurred when reading the IDocs from SAP.

Action: Run the server with RFC trace, resend the IDoc and check the trace file for more details about the error.

SMQ4151 ** Terminating outbound server **

Explanation: The IDoc cannot be processed because the outbound server is ending.

Action: See other messages to find out why the server is ending.

SMQ4152 No inbound queue name was specified.

Explanation: An inbound MQSeries queue name must be specified on the command line or in the ini file.

Action: Re-run the server specifying an inbound queue name.

SMQ4153 No Bad Message queue name was specified.

Explanation: A Bad Message queue name must be specified in the ini file if the terminateonbadmessage parameter is set to N.

Action: Re-run the server specifying a Bad Message queue name.

SMQ4154 The attempt to open the inbound queue failed with reason code <reason code>.

Explanation: The MQOPEN attempt on the inbound queue <queue name> failed with MQSeries reason code <reason code>.

Action: Check MQSeries reason code <reason code>, correct the error and re-run the server.

SMQ4155 The attempt to open the bad message queue failed with reason code <reason code>.

Explanation: The MQOPEN attempt on the bad message queue <queue name> failed with MQSeries reason code <reason code>.

Action: Check MQSeries reason code <reason code>, correct the error and re-run the server.

SMQ4156 **The MQGET from the inbound queue failed with reason code <reason code>.**

Explanation: The attempt to get a message from the inbound message queue failed with reason code <reason code>. The inbound server will end.

Action: Fix the error and restart the inbound server.

SMQ4157 **The server received a termination request from the user exit.**

Explanation: An exit program that was called by the server returned SMQRC_TERMINATE_SERVER so the server will be terminated. Any in-progress transactions will be aborted.

Action: None.

SMQ4158 **Failed to connect to SAP system for inbound IDoc. Check host name and system number.**

Explanation: The RfcOpen call failed for an IDoc on the inbound queue using hostname <host name> and system number <number>. Check that the host name and system number are valid. These are taken from the IDoc or from the initialization file.

Action: None.

SMQ4159 **The client is missing from logon information in IDoc or ini file.**

Explanation: There is no client specified in the logon information for the IDoc and no default client has been specified in the ini file.

Action: Ensure that the RFC destination is set up correctly in SAP (transaction SM59) or supply a default client in the inbound ini file.

SMQ4160 **Language is missing from logon information in IDoc or ini file.**

Explanation: There is no Language specified in the logon information for the IDoc and no default language has been specified in the ini file.

Action: Ensure that the RFC destination is set up correctly in SAP (transaction SM59) or supply a default language in the inbound ini file.

SMQ4161 **Host name missing from logon information in IDoc or ini file.**

Explanation: There is no Host name specified in the logon information for the IDoc and no default host has been specified in the ini file.

Action: Ensure that the RFC destination is set up correctly in SAP (transaction SM59) or supply a default host name in the inbound ini file.

SMQ4162 **User ID missing from logon information in IDoc or ini file.**

Explanation: There is no User ID specified in the logon information for the IDoc and no default user has been specified in the ini file.

Action: Ensure that the RFC destination is set up correctly in SAP (transaction SM59) or supply a default User ID in the inbound ini file.

SMQ4163 Password missing from logon information in IDoc or ini file.

Explanation: There is no password specified in the logon information for the IDoc and no default password has been specified in the ini file.

Action: Ensure that the RFC destination is set up correctly in SAP (transaction SM59) or supply a default password in the inbound ini file.

SMQ4164 System number missing from logon information in IDoc or ini file.

Explanation: There is no System number specified in the logon information for the IDoc and no default system number has been specified in the ini file.

Action: Ensure that the RFC destination is set up correctly in SAP (transaction SM59) or supply a default system number in the inbound ini file.

SMQ4165 Warning on MQGET from inbound queue. Reason code <reason code>.

Explanation: The attempt to get a message from the inbound message queue returned a warning. The warning code was <reason code>. The message will be passed to the user exit if one has been specified. If no exit was specified, the message will be passed to the bad message queue. If no bad message queue was specified, the message will be left on the inbound queue, and the server will terminate.

Action: None.

SMQ4166 IDoc has an invalid structure header. IDoc value="<value 1>". Expected value="<value 2>".

Explanation: The value of the structure Id field in the IDoc header structure contains an invalid value.

Action: Ensure that the IDoc message starts with a valid version of the Saplink header structure (MQSAPH).

SMQ4167 IDoc has an invalid structure version. IDoc value="<value 1>". Expected value="<value 2>".

Explanation: The value of the version field in the IDoc header structure contains an invalid value.

Action: Ensure that the IDoc message starts with a valid version of the Saplink header structure (MQSAPH).

SMQ4168 IDoc has an invalid system number. IDoc value=<value 1>.

Explanation: The value of the system number field in the IDoc header structure contains an invalid system number. The system number specified in the header structure should contain only digits or blanks.

Action: Ensure that the IDoc message starts with a valid version of the Saplink header structure

(MQSAPH). If the IDoc was generated by SAP, check that the system number in the RfcDestination is valid.

SMQ4169 IDoc has an invalid client. IDoc value=<value 1>.

Explanation: The value of the client field in the IDoc header structure contains an invalid client. The client specified in the header structure should contain only digits.

Action: Ensure that the IDoc message starts with a valid version of the Saplink header structure (MQSAPH). If the IDoc was generated by SAP, check that the client in the RfcDestination is valid.

SMQ4170 IDoc has an invalid language. IDoc value=<value 1>.

Explanation: The value of the language field in the IDoc header structure contains an invalid language. The language specified in the header structure should contain only a character or a blank.

Action: Ensure that the IDoc message starts with a valid version of the Saplink header structure (MQSAPH). If the IDoc was generated by SAP, check that the language in the RfcDestination is valid.

SMQ4171 Failed to get SAP transaction ID. Check Client, User Id, Password and Language.

Explanation: The request for a transaction Id from SAP failed. This may be because the logon information in the IDoc or the ini file is not valid or because the connection to R3 has been lost. To get more information, turn on the RFC trace option in the ini file, re-run the server and check the RFC trace (dev_rfc) file.

Action: None.

SMQ4172 The length of the inbound message is invalid for an IDoc.

Explanation: The inbound message data length is not compatible with the IDoc format. The message should be comprised of: The Saplink header structure; followed by an IDoc control structure and IDoc data records for each IDoc in the message. file.

Action: Ensure that IDocs have the correct format.

SMQ4173 The server is terminating because an invalid message was received.

Explanation: See previous messages to find out why the message is invalid.

Action: To prevent the server from terminating when bad messages are received, set the terminateonbadmessage parameter in the ini file to N.

SMQ4174 Inbound IDoc number <number> received from queue.

Explanation: None

Action: None.

SMQ4175 Outbound IDoc number <number> received from SAP.

Explanation: None

Action: None.

SMQ4176 The value is too long. Max. length is 48. (Error in value: <value>)

Explanation: The command line argument is too long.

Action: Change the value specified on the command line and re-run the server.

SMQ4177 Options must begin with '-' and contain no spaces. (Error in: <option>)

Explanation: An invalid option was specified on the command line.

Action: Change the value specified on the command line and re-run the server.

SMQ4178 No initialization file name was specified. This is required.

Explanation: You must specify the name of the initialization file.

Action: Specify the initialization file on the command line and re-run the server.

SMQ4179 Invalid option. Valid options are: '-q', '-m', and '-i'. (Error in:<option>)

Explanation: The valid command line options are:

'-i<ini file name>' (required);

'-q<queue name>', and

'-m<queue manager name>.

Action: Change the values specified on the command line and re-run the server.

SMQ4180 Invalid option. Valid options are: -a,-g,-x,-m and -i. (Error in:<option>)

Explanation: The valid command line options are:

'-i<ini file name>',

'-a<program ID>',

'-g<gateway host>',

'-x<gateway service>', and

'-m<queue manager name>.

Action: Change the values specified on the command line and re-run the server.

SMQ4181 Options -a, -g, and -x must be specified if there is no ini file.

Explanation: You must specify options -a, -g, and -x if there is no ini file.

Action: Change the value specified on the command line and re-run the server.

SMQ4182 Sapl原因 FFST 在进度中。

Explanation: An internal error has occurred and data is being dumped to an FDC file.

Action: Save the error log and FDC dump files and contact your IBM representative.

SMQ4183 一个 IDoc 没有被放入消息队列，参见以前的错误原因。

Explanation: None.

Action: None.

SMQ4184 执行 smqUserExitReturn 在用户退出 <user exit name> 失败。

Explanation: The IDoc failed because an error occurred when the return function of the user exit was called.

Action: Investigate the cause of the failure, fix the error and resend the IDoc.

SMQ4185 用户退出 <user exit name> 失败以转换消息。事务将被回滚。

Explanation: The user exit returned SMQ_CONVERT_FAIL. This indicated it was not able to convert the message.

Action: Investigate the user exit code and the IDoc conversion functions.

SMQ4186 一个无效的结果 (<result code>) 已经从用户退出返回函数接收。

Explanation: The user exit returned an invalid result for this call. Valid result codes are: SMQRC_OK, SMQRC_TERMINATE_SERVER, or, SMQRC_TERMINATE_EXIT.

Action: Change the user exit Return function so that it returns valid result codes.

SMQ4187 尝试将消息放入 SAP 失败。

Explanation: The call to RfcIndirectCall for INBOUND_IDOC_PROCESS failed.

Action: For more information re-run the server with trace and RFC trace enabled.

SMQ4188 原因代码 <reason code> 在打开队列 <queue name> 时。

Explanation: The MQOPEN call to queue <queue name> failed. The return code from the MQOPEN call was <reason code>.

Action: Investigate the MQSeries error code, fix the error and retry the transaction.

SMQ4189 入站队列必须与坏消息队列不同。

Explanation: You have specified the same MQSeries queue for inbound messages and bad messages. This is not allowed.

Action: Change the name of one of the queues and rerun the server.

SMQ4190 The IDoc is larger than the size specified in the ini file.

Explanation: The IDoc or IDoc package is larger than the size specified in the maxidocsize parameter in the ini file.

Action: Increase the maxidocsize parameter in the ini file, or send IDocs in smaller packages.

SMQ4191 A message was put to the bad message queue. Bad message type <number>, reason <reason code>.

Explanation: The message was not in a valid IDoc format. The bad message type is <number> and the bad message reason is <reason code>.

Action: Check the bad message reason code in the bad message header of the message. Attempt to correct the error and send the message again.

SMQ4192 A data conversion problem occurred on the MQGET. Attempting to process message.

Explanation: A warning was issued because the message needs codepage conversion, but the message is either not in MQSTR format, or a user-defined data-conversion exit call failed.

Action: Ensure that incoming messages from machines with a different code page, are in MQSTR format or that there is a user exit defined to convert messages in other formats.

SMQ4193 The exit requested termination. If a transaction was in progress it will be aborted.

Explanation: An exit program that was called by the server returned SMQRC_TERMINATE_EXIT so the exit will be terminated, and any transactions in progress will be aborted.

Action: None.

SMQ4194 An invalid result (<result code>) has been received from the user exit Execute function.

Explanation: The user exit returned an invalid result for this call. Valid result codes are: SMQRC_OK, SMQRC_TERMINATE_SERVER, SMQRC_TERMINATE_EXIT, SMQRC_CONVERT_OK, SMQRC_CONVERT_FAIL, SMQRC_CONVERT_NOT_NEEDED, or, SMQRC_CONVERT_BADMESSAGE.

Action: Change the user exit Execute function so that it returns valid result codes.

SMQ4195 An invalid result (<result code>) has been received from the user exit Initialise function.

Explanation: The user exit returned an invalid result for this call. Valid result codes are: SMQRC_OK, SMQRC_TERMINATE_SERVER, or, SMQRC_TERMINATE_EXIT.

Action: Change the user exit Initialise function so that it returns valid result codes.

SMQ4196 The MQPUT to queue <queue name> failed with reason code <reason code>.

Explanation: The attempt to put a message to the Bad Message queue failed with MQSeries reason code <reason code>.

Action: Fix the error and re-run the server.

SMQ4197 <text>

Explanation: The above text details an error that occurred in a SAP R3 Rfc function.

Action: None.

SMQ4198 **The maximum number of connections to R/3 has been exceeded.**

Explanation: The inbound server has attempted to make more connections to R/3 than are allowed.

Action: Increase the maxconnections parameter in the ini file if it was specified. If it was not specified, the default of 256 was used. Specify a higher value than this in the ini file.

SMQ4199 **The transaction queue must be different to the bad message queue.**

Explanation: You have specified the same MQSeries queue for transactions and bad messages. This is not allowed.

Action: Change the name of one of the queues and rerun the server. transaction.

SMQ4203 **** Transaction is already executing ****

Explanation: An attempt has been made from SAP to execute a transaction that is already running

Action:

SMQ4204 **** Unable to Lock the Transaction Queue ****

Explanation: An attempt to lock the Transaction Queue failed.

Action: If running on UNIX ensure the correct permissions have been set for the directory from which the outbound server is executing

Appendix D. Notices

This information was developed for products and services offered in the United States. IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make

Notices

improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England
SO21 2JN.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Trademarks

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

AIX	AS/400	BookManager
CICS	FFST	First Failure Support Technology
IBM	IBMLink	MQ
MQSeries	MQSeries Three Tier	OS/2
OS/400		

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

R/2, R/3, and SAP are registered trademarks of SAP AG.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Windows and Windows NT are trademarks or registered trademarks of Microsoft Corporation.

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

Glossary of terms and abbreviations

This glossary describes terms used in this book and words used with other than their everyday meaning. In some cases, a definition may not be the only one applicable to a term, but it gives the particular sense in which the word is used in this book.

If you do not find the term you are looking for, see the Index or the *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

ABAP/4. The 4th generation SAP programming language.

application server. An R/3 term for a system on which one or more applications are running. Applications may be R/3 applications.

aRFC. Asynchronous remote function call (an R/3 term)

bad message queue. A queue used by the inbound server to store incoming MQSeries messages when the messages cannot be decomposed into valid IDocs.

exit handler. A subcomponent of the inbound (or outbound) server that calls a user-written program, as specified in the ini file for that server. See also *user exit*.

First Failure Support Technology (FFST). Used by some members of the MQSeries family of products to detect and report software problems.

gateway server. An R/3 gateway through which R/3 applications and SAP link communicate.

local queue. An MQSeries queue that belongs to the local queue manager. A local queue can contain one or more MQSeries messages waiting to be processed.

IDoc. Intermediate document.

A data container used by R/3 applications to send and receive information. MQSeries link for R/3 works only with IDocs.

inbound message queue. An MQSeries queue from which the inbound server takes messages for processing and passes them to the R/3 system.

inbound server. The component of MQSeries link for R/3 that receives MQSeries messages from a specified queue and converts them to IDocs that can be used by R/3.

inbound transaction id queue. A queue used by the inbound server to store transaction IDs associated with an SAP transaction. These IDs are used for backing out transactions that occur within a syncpoint, if a unit of work needs to be rolled back.

ini file. See *initialization file*.

initialization file. A file from which an MQSeries link for R/3 server (inbound or outbound) takes data when it is started.

message. In message queuing applications, a communication sent from one application or program to another.

outbound message queue. An MQSeries queue on which the outbound server puts messages containing IDoc data originating from the R/3 system.

outbound server. The component of MQSeries link for R/3 that receives one or more IDocs from an R/3 system and converts them into an MQSeries message, which it puts on a specified MQSeries queue.

outbound transaction id queue. A queue used by the outbound server to store transaction IDs associated with an R/3 transaction. These IDs are

Glossary

used for backing out transactions that occur within a syncpoint, if a unit of work needs to be rolled back.

remote queue. An MQSeries object, belonging to the local queue manager, that identifies a local queue on another queue manager.

RFC. remote function call (R/3 term)

SM59. An R/3 transaction code that invokes the destinations panels in R/3 so that you can configure R/3 destinations and MQSeries queuing options for the MQSeries link for R/3.

transaction queue. See *inbound transaction id queue* and *outbound transaction id queue*.

translator tools. Programs that can translate IDocs from one format to another. These programs can be called by a user exit.

user exit. A user written program that processes data being transferred through MQSeries link for R/3. The user exit is called by the exit handler on either an inbound or an outbound server.

Bibliography

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

MQSeries cross-platform publications

Most of these publications, which are sometimes referred to as the MQSeries “family” books, apply to all MQSeries Level 2 products. The latest MQSeries Level 2 products are:

- MQSeries for AIX, V5.2
- MQSeries for AS/400, V5.2
- MQSeries for AT&T GIS UNIX, V2.2
- MQSeries for Compaq (DIGITAL) OpenVMS, V2.2.1.1
- MQSeries for Compaq Tru64 UNIX, V5.1
- MQSeries for HP-UX, V5.2
- MQSeries for Linux, V5.2
- MQSeries for OS/2 Warp, V5.1
- MQSeries for OS/390, V5.2
- MQSeries for SINIX and DC/OSx, V2.2
- MQSeries for Sun Solaris, V5.2
- MQSeries for Sun Solaris, Intel Platform Edition, V5.1
- MQSeries for Tandem NonStop Kernel, V2.2.0.1
- MQSeries for VSE/ESA, V2.1.1
- MQSeries for Windows, V2.0
- MQSeries for Windows, V2.1
- MQSeries for Windows NT and Windows 2000, V5.2

The MQSeries cross-platform publications are:

- *MQSeries Brochure*, G511-1908
- *An Introduction to Messaging and Queuing*, GC33-0805
- *MQSeries Intercommunication*, SC33-1872
- *MQSeries Queue Manager Clusters*, SC34-5349

- *MQSeries Clients*, GC33-1632
- *MQSeries System Administration*, SC33-1873
- *MQSeries MQSC Command Reference*, SC33-1369
- *MQSeries Event Monitoring*, SC34-5760
- *MQSeries Programmable System Management*, SC33-1482
- *MQSeries Administration Interface Programming Guide and Reference*, SC34-5390
- *MQSeries Messages*, GC33-1876
- *MQSeries Application Programming Guide*, SC33-0807
- *MQSeries Application Programming Reference*, SC33-1673
- *MQSeries Programming Interfaces Reference Summary*, SX33-6095
- *MQSeries Using C++*, SC33-1877
- *MQSeries Using Java™*, SC34-5456
- *MQSeries Application Messaging Interface*, SC34-5604

MQSeries platform-specific publications

Each MQSeries product is documented in at least one platform-specific publication, in addition to the MQSeries family books.

MQSeries for AIX, V5.2

MQSeries for AIX Quick Beginnings, GC33-1867

MQSeries for AS/400, V5.2

MQSeries for AS/400® Quick Beginnings, GC34-5557

Bibliography

*MQSeries for AS/400 System
Administration, SC34-5558*

*MQSeries for AS/400 Application
Programming Reference (ILE RPG),
SC34-5559*

MQSeries for AT&T GIS UNIX, V2.2

*MQSeries for AT&T GIS UNIX[®]
System Management Guide,
SC33-1642*

MQSeries for Compaq (DIGITAL)

OpenVMS, V2.2.1.1

*MQSeries for Compaq (DIGITAL)
OpenVMS System Management
Guide, GC33-1791*

MQSeries for Compaq Tru64 UNIX, V5.1

*MQSeries for Compaq Tru64 UNIX
Quick Beginnings, GC34-5684*

MQSeries for HP-UX, V5.2

*MQSeries for HP-UX Quick
Beginnings, GC33-1869*

MQSeries for Linux, V5.2

*MQSeries for Linux Quick
Beginnings, GC34-5691*

MQSeries for OS/2 Warp, V5.1

MQSeries for OS/2 Warp Quick Beginnings, GC33-1868

MQSeries for OS/390, V5.2

MQSeries for OS/390 Concepts and Planning Guide, GC34-5650

MQSeries for OS/390 System Setup Guide, SC34-5651

MQSeries for OS/390 System Administration Guide, SC34-5652

MQSeries for OS/390 Problem Determination Guide, GC34-5892

MQSeries for OS/390 Messages and Codes, GC34-5891

MQSeries for OS/390 Licensed Program Specifications, GC34-5893

MQSeries for OS/390 Program Directory

MQSeries link for R/3, Version 1.2

MQSeries link for R/3 User's Guide, GC33-1934

MQSeries for SINIX and DC/OSx, V2.2

MQSeries for SINIX and DC/OSx System Management Guide, GC33-1768

MQSeries for Sun Solaris, V5.2

MQSeries for Sun Solaris Quick Beginnings, GC33-1870

MQSeries for Sun Solaris, Intel Platform Edition, V5.1

MQSeries for Sun Solaris, Intel Platform Edition Quick Beginnings, GC34-5851

MQSeries for Tandem NonStop Kernel, V2.2.0.1

MQSeries for Tandem NonStop Kernel System Management Guide, GC33-1893

MQSeries for VSE/ESA, V2.1.1

MQSeries for VSE/ESA™ Licensed Program Specifications, GC34-5365

MQSeries for VSE/ESA System Management Guide, GC34-5364

MQSeries for Windows, V2.0

MQSeries for Windows® User's Guide, GC33-1822

MQSeries for Windows, V2.1

MQSeries for Windows User's Guide, GC33-1965

MQSeries for Windows NT and Windows 2000, V5.2

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

MQSeries for Windows NT® Using the Component Object Model Interface, SC34-5387

MQSeries LotusScript Extension, SC34-5404

Softcopy books

Most of the MQSeries books are supplied in both hardcopy and softcopy formats.

HTML format

Relevant MQSeries documentation is provided in HTML format with these MQSeries products:

- MQSeries for AIX, V5.2
- MQSeries for AS/400, V5.2
- MQSeries for Compaq Tru64 UNIX, V5.1
- MQSeries for HP-UX, V5.2
- MQSeries for Linux, V5.2
- MQSeries for OS/2 Warp, V5.1
- MQSeries for OS/390, V5.2
- MQSeries for Sun Solaris, V5.2
- MQSeries for Sun Solaris, Intel Platform Edition, V5.1
- MQSeries for Windows NT and Windows 2000, V5.2 (compiled HTML)
- MQSeries link for R/3, V1.2

Bibliography

The MQSeries books are also available in HTML format from the MQSeries product family Web site at:

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

Portable Document Format (PDF)

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

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

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

PDF versions of relevant MQSeries books are supplied with these MQSeries products:

- MQSeries for AIX, V5.2
- MQSeries for AS/400, V5.2
- MQSeries for Compaq Tru64 UNIX, V5.1
- MQSeries for HP-UX, V5.2
- MQSeries for Linux, V5.2
- MQSeries for OS/2 Warp, V5.1
- MQSeries for OS/390, V5.2
- MQSeries for Sun Solaris, V5.2
- MQSeries for Sun Solaris, Intel Platform Edition, V5.1
- MQSeries for Windows NT and Windows 2000, V5.2
- MQSeries link for R/3, V1.2

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

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

BookManager® format

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

BookManager READ/2

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

PostScript format

The MQSeries library is provided in PostScript (.PS) format with many MQSeries Version 2 products. Books in PostScript format can be printed on a PostScript printer or viewed with a suitable viewer.

Windows Help format

The *MQSeries for Windows User's Guide* is provided in Windows Help format with MQSeries for Windows, Version 2.0 and MQSeries for Windows, Version 2.1.

MQSeries information available on the Internet

The MQSeries product family Web site is at:

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

By following links from this Web site you can:

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

Related publications

The following books which are available from SAP may also be useful:

ALE Programmers Guide
ALE Consultants Manual

Index

A

accessing OS/390 UNIX services 30
AIX, uninstalling R/3 20
AIX directories 19
AIX installation 15
application data, in messages 55
application link enabling (ALE) 1
application server field 38

B

bad message queue 25
bad messages
 error type 88
 handling 86
 header 87
 processing 89
 reason code 88
basic configuration, testing 41
batch process for inbound server
 (OS/400) 61
batch process for outbound server
 (OS/400) 64
batch size for IDocs 74
batching, parameter 74
batching IDocs 5
benefits 2
bibliography 131
BookManager 134

C

C code, user exits 45
calling user exits 45
channels, receiver 26
choosing queue types 25
client field 37
commands
 initialization parameters 59
 smqsi (start inbound server) 60
 smqso (start outbound
 server) 62
 start inbound server (smqsi) 60
 start outbound server
 (smqso) 62
communications failures 85
Compaq Tru64 UNIX, uninstalling
 R/3 20
Compaq Tru64 UNIX directories 19
Compaq Tru64 UNIX
 installation 16
compiling user exits 53

configuration
 inbound parameters 106
 MQSeries 26
 outbound parameters 105
 testing 41
configuring
 OS/390 30
connection parameters,
 specifying 37
control data, in IDocs 58
creating a SAP logon group 34

D

data conversion 85
 exit 100
 exit library, MQHSAP 24, 58
 exit problems 85
 failures 85
defaults, user exits 72
defining
 inbound server 25
 MQSeries 73
 MQSeries destination
 parameters 35
 MQSeries objects 23, 31
 outbound server 24
 queue managers 24
 R/3 connection parameters 37
Defining TCP/IP ports 31
destination
 parameters 71
 RFC, specifying 35
 RFC R/3 5, 29, 32
Destination configuration panel 35
directories
 AIX 19
 AS/400 20
 Compaq Tru64 UNIX 19
 error logs 91
 HP-UX 19
 OS/390 20
 Sun 19
 Windows NT 20

E

enabling user exits 47
entry point
 return user exit 52
 terminate user exit 52
 user exit 50

entry point (*continued*)
 user exit, initialize 49
environment
 creating for MQSeries on
 OS/390 40
error
 code, bad message 88
 handling 83
 logs 91
 messages 109
 outbound server 83
 problems starting on
 OS/390 84
example MQSeries
 configurations 26
execute entry point 50
exit, data conversion, problems 85

F

failure, of MQSeries link for R/3 85
files, ini 39
formats, message 55

G

gateway
 inbound server 75
 initializing 39
 outbound server 71
 what they are 71
glossary 129

H

hardware requirements 11
header, bad message 87
host name, Gateway 6
HP-UX, uninstalling R/3 20
HP-UX directories 19
HP-UX installation 16
HTML (Hypertext Markup
 Language) 133
Hypertext Markup Language
 (HTML) 133

I

IDocs
 batching 5
 control data 58
 data structure 55
 monitoring traffic 43
 size of batch 74
 unique identifier 58

- inbound message queue 25
- inbound server batch process (OS/400) 61
- inbound servers 6
 - configuration parameters 106
 - errors 84
 - initialization parameters 75
 - initializing 39
 - load balancing 7
 - load balancing setup 34
 - mapping to SAP logon group 35
 - running as Windows NT service 67
 - sequence of events 6
- ini files
 - host 71
 - host name 6
 - saprfc 8, 59, 80
 - service, ini file 72
 - specifying server configuration 39
- initialization
 - files 39
 - inbound server parameters 75
 - outbound server parameters 71
 - sample files 95
- initializing
 - inbound server 39
 - outbound server 39
- installation
 - AIX 15
 - Compaq Tru64 UNIX 16
 - HP-UX 16
 - OS/390 17
 - OS/400 17
 - review READ.ME first 15
 - software requirements 11
 - Sun Solaris 17
 - Windows NT 18
- introduction 1
- L**
 - language field 37
 - load balancing 7, 40
 - load balancing setup 34
 - logging, parameter 74, 80
 - logon group
 - SAP, creating 34
 - SAP, mapping to inbound server 35
 - logs, error 91
- M**
 - mapping inbound server to SAP 35
 - maximum size, IDoc batch 74

- Message Queuing destination
 - configuration 35
 - message size restrictions 12
 - messages
 - bad 84, 86, 88
 - error 109
 - error type 88
 - formats 55
 - MQSeries link for R/3 header structure 56
 - reason code 88
 - unrecognizable 84, 86
 - MQHSAP
 - data conversion exit library 24, 58
 - setting format field 55
 - MQSC command files 102
 - MQSeries
 - command files 102
 - configuration 26
 - defining objects 23, 31
 - ini file definitions 73
 - mapping R/3 logical destinations 35
 - object definitions for an inbound server 25
 - object definitions for an outbound server 24
 - platforms supported 12
 - MQSeries publications 131
 - multiple outbound servers 5
 - N**
 - name, gateway host 6
 - O**
 - object, remote queue 27
 - operating systems 12
 - options file field 38
 - Options panel, Message Queuing 35
 - OS/390
 - accessing UNIX services 30
 - configuring 30
 - Creating the MQSeries environment 40
 - data conversion exit 101
 - problems starting the outbound server 84
 - sample JCL 103
 - UNIX shell 30
 - OS/390, uninstalling R/3 21
 - OS/390 directories 20
 - OS/390 installation 17
 - OS/400, uninstalling R/3 21
 - OS/400 directories 20

- OS/400 installation 17
- outbound queue, choosing 25
- outbound server
 - batching IDocs 5
 - configuration parameters 105
 - errors 83
 - starting the outbound server on OS/390 84
 - initialization parameters 71
 - initializing 39
 - MQSeries definitions for 24
 - queue name for 36
 - RFC destination parameters 71
 - running as Windows NT service 67
 - running multiple 5
 - transaction queue name 73
- outbound server batch process (OS/400) 64
- outbound servers 3
- outbound user exit defaults 72
- overview 1
- P**
 - parameters
 - inbound configuration 106
 - initialization 59
 - MQSeries destination 35
 - outbound configuration 105
 - password
 - field 37
 - for MQSeries 93
 - PDF (Portable Document Format) 134
 - platforms
 - for MQSeries 12
 - for MQSeries link for R/3 12
 - Portable Document Format (PDF) 134
 - PostScript format 134
 - problems 83
 - product overview 1
 - program ID, ini file 72
 - publications
 - MQSeries 131
- Q**
 - queue manager
 - defining 24
 - name in ini file 73
 - specifying a name 37
 - starting on OS/390 30
 - starting on Windows NT 67
 - using more than one 28
 - queues
 - for an outbound server 36

queues (*continued*)

- outbound 25
- remote 27
- transmission 25

quick reference 105

R

R/3 connection parameters 37

R/3 destinations 29, 32

README file 15

reason code, bad message 88

receiver channel 26

reference, quick 105

remote queue object 27

removing R/3, AIX 20

removing R/3, Compaq Tru64

UNIX 20

removing R/3, HP-UX 20

removing R/3, OS/390 21

removing R/3, OS/400 21

removing R/3, Sun Solaris 20

removing R/3, Windows NT 21

requirements

hardware 11

software 11

return (user exit entry point) 52

RFC

destination parameters 71

destinations 5, 29, 32

rolling back a unit of work 85

S

samples

directory 53

ini files 95

JCL for OS/390 103

MQSC command file 26

MQSC command files 102

user exits 53, 99, 100

SAP

creating logon group 34

link failure 85

link header structure 56

mapping logon group to

server 35

specifying an R/3 connection 37

SAPGUI component of R/3 11

saprfc.ini file 8, 40, 80

SASQASM on OS/390 103

SASQJCL on OS/390 31, 40, 103

SBMJOB command 61, 64

security 93

server

inbound 3, 6

inbound, parameters 60

inbound, starting 60

server (*continued*)

inbound, stopping 62

initializing 39

MQSeries link for R/3,

understanding 3

outbound 3, 62

outbound, parameters 62

outbound, starting 62

outbound, stopping 65

running as Windows NT

services 67

starting 40

sm59 6, 33

smqDestConf file 28, 29, 35, 39, 95

smqsi, start inbound server 60

smqso, start outbound server 62

softcopy books 133

software requirements 11

starting

inbound server 60

outbound server 62

starting servers 40

stopping

inbound server 60

outbound server 62

Sun directories 19

Sun Solaris, uninstalling R/3 20

Sun Solaris installation 17

supported platforms 12

SupportPac 134

system connection defaults 75

system ID field 38

T

TCP/IP ports, defining 31

terminate (user exit entry point) 52

testing, basic configuration 41

trace, turning on 73, 79

trace files in MQSeries link for

R/3 90

trademarks 127

transaction queue name 73

transmission queue 25

troubleshooting 83

tst files, MQSC commands 102

type, outbound queue 25

U

understanding MQSeries link for

R/3 servers 3

uninstalling R/3, AIX 20

uninstalling R/3, Compaq Tru64

UNIX 20

uninstalling R/3, HP-UX 20

uninstalling R/3, OS/390 21

uninstalling R/3, OS/400 21

uninstalling R/3, Sun Solaris 20

uninstalling R/3, Windows NT 21

unique IDoc identifier 58

unit of work, rolling back 85

UNIX services

accessing on OS/390 30

BPXBATCH on OS/390 30

OMVS on OS/390 30

rlogin on OS/390 30

unrecognizable message 84, 86

user exit

buffer 37

compiling 53

data conversion 100

defaults 72

enabling 37, 47

execute entry point 50

for MQSeries link for R/3 45

handler 3

ini file buffer 73

ini file name 78

initialize entry point 49

name 72

overview 8

return entry point 52

samples 53, 99, 100

specifying a name 37

terminate entry point 52

when called 45

writing 45

user ID 93

field 37

W

Windows Help 134

Windows NT, uninstalling R/3 21

Windows NT directories 20

Windows NT installation 18

Windows NT services 67

writing user exits 45

Sending your comments to IBM

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

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

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

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

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

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

- By mail, to this address:
User Technologies Department (MP095)
IBM United Kingdom Laboratories
Hursley Park
WINCHESTER,
Hampshire
SO21 2JN
United Kingdom
- By fax:
 - From outside the U.K., after your international access code use 44-1962-870229
 - From within the U.K., use 01962-870229
- Electronically, use the appropriate network ID:
 - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
 - IBMLink™: HURSLEY(IDRCF)
 - Internet: idrcf@hursley.ibm.com

Whichever method you use, ensure that you include:

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



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

GC33-1934-04



Spine information:



MQSeries

MQSeries link for R/3 User's Guide

Version 1.2