WebSphere MQ for z/OS

Problem Determination Guide

Version 7.0

GC34-6944-01

WebSphere MQ for z/OS

Problem Determination Guide

Version 7.0

GC34-6944-01

Note
Before using this information and the product it supports, be sure to read the general information under notices at the back of this book.

Second edition (January 2009)

This edition of the book applies to the following products:

- IBM WebSphere MQ for Windows®, Version 7.0

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

Contents

Figures v

Tables vii

Chapter 1. Establishing the cause of the problem 1
Introduction to problem determination 1
How this book can help you 1
Preliminary checks 2
Has WebSphere MQ for z/OS run successfully before? 3
Are there any error messages or return codes that explain the problem? 3
Can you reproduce the problem? 4
Have any changes been made since the last successful run? 4
Has the application run successfully before? . . . 5
Does the problem affect specific parts of the network? 6
Does the problem occur at specific times of the day? 7
Does the problem affect all users of the application?. 7
Does the problem occur with all z/OS, CICS, or IMS systems? 7
Is the problem intermittent? 7
Have you applied any APARs or PTFs? 8
Do you have a program error? 8
What to do next 9
Examining the problem in greater depth 9
Have you obtained some incorrect output? . . . 9
Have you received an unexpected error message? 10
Has there been an abend? 10
Have you failed to receive a response from an MQSC command? 11
Is there a problem with the WebSphere MQ queues? 13
Is your application or WebSphere MQ for z/OS running slowly? 16
Has your application or WebSphere MQ for z/OS stopped processing work? 17
Where to look next 18

Chapter 2. Dealing with the problem . . 21
Dealing with program abends 21
Batch abends 21
CICS transaction abends 21
IMS transaction abends 22
Dealing with waits and loops 22
Distinguishing between waits and loops 22
Dealing with waits 23
Dealing with loops 25
Dealing with performance problems 27
z/OS system considerations 28
WebSphere MQ for z/OS considerations 28
CICS constraints 30
Application design considerations 31
Dealing with incorrect output 33
Messages do not appear when expected 33
Messages contain unexpected or corrupted information 38

Chapter 3. Diagnostic aids and techniques 41
Diagnostic aids 41
WebSphere MQ for z/OS recovery actions . . . 41
WebSphere MQ for z/OS abends 42
Diagnostic information produced 44
Other sources of information 46
Diagnostic aids for CICS 47
Diagnostic aids for IMS 47
Diagnostic aids for DB2 48
WebSphere MQ dumps 48
How to use dumps for problem determination . 48
Getting a dump 48
Processing a dump 51
Analyzing the dump 61
SYSUDUMP information 62
Snap dumps 63
SYS1.LOGREC information 64
When SVC dumps are not produced 64
Using trace for problem determination 65
The user parameter and IBM internal traces . . 65
The Log and Trace Analyzer tool 70
Examples of trace output 70
Other types of trace 72

Chapter 4. Finding solutions to similar problems 75
Searching the IBM database 75
Search argument process 75
The keyword format 76
Building a keyword string 78
Component-identifier keyword 81
Abend keyword 82
Wait and loop keywords 85
Message keyword 85
Performance keyword 87
Documentation keyword 88
Incorrect output keyword. 89

Chapter 5. Working with IBM to solve your problem 91
Reporting a problem to the IBM software support group 91
When to contact the support center 91
Dealing with the support center 91
Collecting documentation for the problem . . . 95
Sending the documentation to the change team 96

Resolving a problem 97
The APAR process 97
Applying the fix 97

Chapter 6. SDB format symptom-to-keyword cross reference . 99

Chapter 7. WebSphere MQ component and resource manager identifiers . . . 101

Chapter 8. CICS adapter trace entries 103

Chapter 9. Examples of CEDF output 107

MQOPEN 107
MQCLOSE 108
MQPUT 109
MQPUT1 111
MQGET 112
MQINQ 113
MQSET 115

Notices 117

Index 119

Sending your comments to IBM . . . 123

Figures

1. Sample symptom string 45
2. Dumping the WebSphere MQ queue manager and application address spaces 50
3. Dumping the WebSphere MQ queue manager address space 50
4. Dumping the channel initiator address space 50
5. Dumping the WebSphere MQ queue manager and channel initiator address spaces 50
6. Dumping a Coupling Facility structure 51
7. Sample JCL for printing dumps through IPCS in the z/OS environment 60
8. Sample SVC dump title 61
9. Dump title with PSW and ASID. 62
10. Sample beginning of a SYSUDUMP 63
11. Example startup of GTF to use with the WebSphere MQ trace 66
12. Example of GTF Stop command to use with the WebSphere MQ trace 67
13. Formatting the GTF output in batch 68
14. Example trace data from an entry trace of an MQPUT1 request 71
15. Example trace data from an exit trace of an MQPUT1 request 72
16. High-level flowchart of various sets of keywords 80
17. Sample problem reporting sheet. 92
18. Example CEDF output on entry to an MQOPEN call (hexadecimal) 107
19. Example CEDF output on exit from an MQOPEN call (hexadecimal) 108
20. Example CEDF output on entry to an MQOPEN call (character) 108
21. Example CEDF output on exit from an MQOPEN call (character) 108
22. Example CEDF output on entry to an MQCLOSE call (hexadecimal) 108
23. Example CEDF output on exit from an MQCLOSE call (hexadecimal) 109
24. Example CEDF output on entry to an MQCLOSE call (character) 109
25. Example CEDF output on exit from an MQCLOSE call (character) 109
26. Example CEDF output on entry to an MQPUT call (hexadecimal) 110
27. Example CEDF output on exit from an MQPUT call (hexadecimal) 110
28. Example CEDF output on entry to an MQPUT call (character) 110
29. Example CEDF output on exit from an MQPUT call (character) 110
30. Example CEDF output on entry to an MQPUT1 call (hexadecimal). 111
31. Example CEDF output on exit from an MQPUT1 call (hexadecimal). 111
32. Example CEDF output on entry to an MQPUT1 call (character) 111
33. Example CEDF output on exit from an MQPUT1 call (character) 112
34. Example CEDF output on entry to an MQGET call (hexadecimal) 112
35. Example CEDF output on exit from an MQGET call (hexadecimal) 113
36. Example CEDF output on entry to an MQGET call (character) 113
37. Example CEDF output on exit from an MQGET call (character) 113
38. Example CEDF output on entry to an MQINQ call (hexadecimal) 114
39. Example CEDF output on exit from an MQINQ call (hexadecimal) 114
40. Example CEDF output on entry to an MQINQ call (character) 114
41. Example CEDF output on exit from an MQINQ call (character) 115
42. Example CEDF output on entry to an MQSET call (hexadecimal) 115
43. Example CEDF output on exit from an MQSET call (hexadecimal) 116
44. Example CEDF output on entry to an MQSET call (character) 116
45. Example CEDF output on exit from an MQSET call (character) 116

Tables

1. Where to look next 18
2. Abend completion codes 42
3. Types of dump used with WebSphere MQ for z/OS. 49
4. Keywords for the WebSphere MQ for z/OS dump formatting control statement 56
5. Resource manager dump formatting keywords 57
6. Summary dump keywords for the WebSphere MQ for z/OS dump formatting control statement 58
7. IPCS subcommands used for dump analysis 58
8. Keywords for the IPCS VERBEXIT CSQXDPRD 59
9. Control blocks traced for WebSphere MQ MQI calls 69
10. Keywords defined for use in a free format search 77
11. Keywords defined for use in a free format search 77
12. Types of WebSphere MQ for z/OS failures 81
13. Message prefixes 86
14. INCORROUT modifier keywords: RECOVERY 89
15. INCORROUT modifier keywords: UTILITY 89
16. SDB format symptom-to-keyword cross-reference 99
17. WebSphere MQ component and resource manager identifiers 101
18. CICS adapter trace entries 103

Chapter 1. Establishing the cause of the problem

Introduction to problem determination

This book tells you how to find the reasons for problems with WebSphere® MQ for z/OS®. Usually, you start with a symptom, or set of symptoms, and trace them back to their cause. This book describes tools and techniques you can use to find the cause of a problem and suggests actions for solving the problem.

The process of problem determination often enables you to solve a problem. For example:

- If you find that the cause of the problem is an error in an application program, you can solve the problem by fixing the error.
- If you find that the cause of the problem is an error in your system programming (such as an error in resource definition), you can solve the problem by correcting the error.

However, you might not always be able to solve a problem after determining its cause. For example:

- A performance problem might be caused by a limitation of your hardware.
- You might find that the cause of the problem is in WebSphere MQ for z/OS itself. If this happens, you need to contact your IBM® support center for a solution.

How this book can help you

The approach in this book is to start with the symptoms of the problem, and try to use these symptoms to classify it. For each class of problem, possible causes are suggested, together with techniques you can use to establish what the cause is.

It is always assumed that the problem has a simple cause to start with (for example, an application programming error). If, as a result of investigation, it becomes clear that the cause of the problem is not straightforward, you must consider possible causes that might be more difficult to determine (and solve). Having exhausted all other possibilities, consider the possibility that the cause of the problem might be in WebSphere MQ itself, in which case you would need to get help from IBM.

How this book is organized

This book contains the following sections:

Part 1. Establishing the cause of the problem
This section outlines the simple preliminary checks and tests that you should perform to try and establish the cause of the problem. You should find that working through this section is all that is required to solve your problem.

Part 2. Dealing with the problem
This section tells you how to deal with more complex WebSphere MQ problems, diagnosed in the previous section. It covers the following problems:

- Program abends
- Waits and loops
- Performance problems
- Incorrect output

Part 3. Diagnostic aids and techniques
This section covers the tools you can use in solving your WebSphere MQ problems. It discusses the use of dumps and traces, and gives a list of other possible sources of information about the problem.

Part 4. Finding solutions to similar problems
This section tells you how to develop a set of keywords to search the IBM software support database for solutions to similar problems.

Part 5. Working with IBM to solve your problem
This section tells you what to expect if you need to contact the IBM support center for help with your problem. It also covers the APAR process.

Preliminary checks

Before you start problem determination in detail, it is worth considering the facts to see if there is an obvious cause of the problem, or a likely area in which to start your investigation. This approach to debugging can often save a lot of work by highlighting a simple error, or by narrowing down the range of possibilities.

The cause of your problem could be in any of the following:

- Queue manager address space
- A queue manager in your queue-sharing group
- Channel initiator address space
- CICS® address space
- IMS™ region
- Batch or TSO address space
- The z/OS system (including ARM, RRS, or the Coupling Facility)
- The network (including APPC or TCP/IP)
- Another system, for example a queue manager on another platform, or a WebSphere MQ client
- The external security manager product, for example RACF® or ACF2
- DB2®

The sections that follow raise some fundamental questions that you need to consider.

As you go through the questions, make a note of anything that might be relevant to the problem. Even if your observations do not suggest a cause straightaway, they could be useful later if you have to carry out a systematic problem determination exercise.

The following list summarizes the preliminary checks that you should make:

- "Has WebSphere MQ for z/OS run successfully before?" on page 3
- "Are there any error messages or return codes that explain the problem?" on page 3
- "Can you reproduce the problem?" on page 4

- "Have any changes been made since the last successful run?" on page 4
- "Has the application run successfully before?" on page 5
- "Does the problem affect specific parts of the network?" on page 6
- "Does the problem occur at specific times of the day?" on page 7
- "Does the problem affect all users of the application?" on page 7
- "Does the problem occur with all z/OS, CICS, or IMS systems?" on page 7
- "Is the problem intermittent?" on page 7
- "Have you applied any APARs or PTFs?" on page 8

Has WebSphere MQ for z/OS run successfully before?

If the answer to this question is **No**, consider the following:

- Check your setup

  If WebSphere MQ has not run successfully on z/OS before, it is likely that you have not yet set it up correctly. See the information about installing and customizing the queue manager in the WebSphere MQ for z/OS System Setup Guide for guidance on doing this.
- Verify the installation
- Check that message CSQ9022I was issued in response to the START QMGR command (indicating normal completion).
- Ensure that z/OS displays WebSphere MQ as an installed subsystem. Use the z/OS command `D OPDATA` to do this.
- Check that the installation verification program (IVP) ran successfully.
- Issue the command `DISPLAY DQM` to check that the channel initiator address space is running, and that the appropriate listeners are started.

Are there any error messages or return codes that explain the problem?

The problem might produce the following types of error message or return codes:

CSQ messages and reason codes
: WebSphere MQ for z/OS error messages have the prefix CSQ; if you receive any messages with this prefix (for example, in the console log, or the CICS log), look in the WebSphere MQ for z/OS Messages and Codes manual for an explanation.

Other messages
: For messages with a different prefix, look in the appropriate messages and codes manual for a suggested course of action that might help you resolve the problem. Table 13 on page 86 lists possible message prefixes and appropriate manuals.

Unusual messages
: Any unusual messages associated with the startup of WebSphere MQ for z/OS, or issued while the system was running before the error occurred, might indicate some system problem that prevented your application from running successfully.

Application MQI return codes
: If your application gets a return code indicating that an MQI call has failed, see the WebSphere MQ Application Programming Reference manual for a description of that return code.

Can you reproduce the problem?

If you can reproduce the problem, consider the conditions under which you can reproduce it. For example:

Is it caused by a command?
If so, is the command issued from the z/OS console, from CSQUTIL, from a program written to put commands onto the SYSTEM.COMMAND.INPUT queue, or by using the operations and control panels?

Does the command work if it is entered by another method?
If the command works when it is entered at the console, but not otherwise, check that the command server has not stopped, and that the queue definition of the SYSTEM.COMMAND.INPUT queue has not been changed.

Is the command server running?
Issue the command `DIS CMDSERV` to check.

Is it caused by an application?

If so, does it fail in CICS, IMS, TSO, or batch?

Does it fail on all WebSphere MQ systems, or only on some?

Is an application causing the problem?
Can you identify any application that always seems to be running in the system when the problem occurs? If so, examine the application to see if it is in error.

Have any changes been made since the last successful run?

When you are considering changes that might recently have been made, think about WebSphere MQ, and also about the other programs it interfaces with, the hardware, and any new applications. Consider also the possibility that a new application that you don't yet know about might have been run on the system.

Has your initialization procedure been changed?
Consider whether that might be the cause of the problem. Have you changed any data sets, or changed a library definition? Has z/OS been initialized with different parameters? In addition, check for error messages sent to the console during initialization.

Have you changed any queue definitions or security profiles?
Consider whether some of your queues have been altered so that they are members of a cluster. This might mean that messages arrive from different sources (for example, other queue managers or applications).

Have you changed any definitions in your sysplex that relate to the support and implementation of shared queues?
Consider the effect that changes to such definitions as your sysplex couple data set, or Coupling Facility resource management policy might have on the operation of shared queues. Also, consider the effect of changes to the DB2 data sharing environment.

Has any of the software on your z/OS system been upgraded to a later release?
Consider whether there are any necessary post-installation or migration activities that you need to perform.

Has your z/OS subsystem name table been changed?

Changes to levels of corequisite software like z/OS or LE might mean that you have to also make some changes to WebSphere MQ.

Do your applications deal with return codes that they might get as a result of any changes you have made?

Ensure that your applications deal with any new return codes that you introduce.

Has the application run successfully before?

If the problem appears to involve one particular application, consider whether the application has run successfully before.

Before you answer **Yes** to this question, consider the following:

Have any changes been made to the application since it last ran successfully?

If so, it is likely that the error lies somewhere in the new or modified part of the application. Take a look at the changes and see if you can find an obvious reason for the problem.

Have all the functions of the application been fully exercised before?

Could it be that the problem occurred when part of the application that had never been invoked before was used for the first time? If so, it is likely that the error lies in that part of the application. Try to find out what the application was doing when it failed, and check the source code in that part of the program for errors.

If a program has been run successfully on many previous occasions, check the current queue status and files that were being processed when the error occurred. It is possible that they contain some unusual data value that causes a rarely used path in the program to be invoked.

Does the application check all return codes?

Could it be that your system has been changed, perhaps in a minor way, but your application does not check the return codes it receives as a result of the change. For example:

- Does your application assume that the queues it accesses can be shared? If a queue has been redefined as exclusive, can your application deal with return codes indicating that it can no longer access that queue?
- Have any security profiles been altered? An **MQOPEN** call could fail because of a security violation; can your application recover from the resulting return code?

Does the application expect particular message formats?

If a message with an unexpected message format has been put onto a queue (for example, a message from a queue manager on a different platform) it might require data conversion or another different form of processing.

Does the application run on other WebSphere MQ for z/OS systems?

Could it be that there is something different about the way that this queue manager is set up that is causing the problem? For example, have the queues been defined with the same maximum message length, or default priority?

Does the application use the MQSET call to change queue attributes?

Could it be that, for example, the application is designed to set a queue to

have no trigger, then process some work, then set the queue to have a trigger? The application might have failed before the queue had been reset to have a trigger.

Does the application handle messages that cause an application to fail?
If an application fails because of a corrupted message, the message retrieved is rolled back. The next application might get the same message and fail in the same way. Ensure that applications use the backout count; when the backout count threshold has been reached, the message in question is put onto the backout queue.

If your application has never run successfully before, examine your application carefully to see if you can find any of the following errors:

Translation and compilation problems
Before you look at the code, examine the output from the translator, the compiler or assembler, and the linkage editor, to see if any errors have been reported. If your application fails to translate, compile/assemble, or link-edit into the load library, it will also fail to run if you attempt to invoke it. See the WebSphere MQ Application Programming Guide for information about building your application, and for examples of the job control language (JCL) statements required.

Batch and TSO programs
For batch and TSO programs, check that the correct stub has been included. There is one batch stub and two RRS stubs. If you are using RRS, check that you are not using the MQCMIT and MQBACK calls with the CSQBRSTB stub; use the CSQBRRSI stub if you want to continue using these calls with RRS.

CICS programs
For CICS programs, check that the program, the WebSphere MQ CICS stub, and the CICS stub have been linked in the correct order. Also, check that your program or transaction is defined to CICS.

IMS programs
For IMS programs, check that the link includes the program, the WebSphere MQ stub, and the IMS language interface module, and that the correct entry point has been specified. Note that a program that is loaded dynamically from an IMS program must have the stub and language interface module linked also if it is to use WebSphere MQ.

Possible code problems
If the documentation shows that each step was accomplished without error, consider the coding of the application. Do the symptoms of the problem indicate the function that is failing and, therefore, the piece of code in error? See “Do you have a program error?” on page 8 for some examples of common errors that cause problems with WebSphere MQ applications.

Do applications report errors from WebSphere MQ?
For example, a queue might not be enabled for ″gets″. It receives a return code specifying this but does not report it. Consider where your applications report any errors or problems.

Does the problem affect specific parts of the network?

You might be able to identify specific parts of the network that are affected by the problem (remote queues, for example). If the link to a remote queue manager is

not working, the messages cannot flow to a target queue on the target queue manager. Check that the connection between the two systems is available, and that the channel initiator and listener have been started. Use the MQSC `PING CHANNEL` command to check the connection.

Check that messages are reaching the transmission queue, and check the local queue definition of the transmission queue, and any remote queues. Use the MQSC `BYTSSENT` keyword of the `DISPLAY CHSTATUS` command to check that data is flowing along the channel. Use `DISPLAY QLOCAL (XMITQ) CURDEPTH` to check whether there are messages to be sent on the transmission queue. Check for diagnostic messages at both ends of the channel informing you that messages have been sent to the dead-letter queue.

If you are using WebSphere MQ clusters, check that the clustering definitions have been set up correctly.

Have you made any network-related changes that might account for the problem?

Have you changed any WebSphere MQ definitions, or any CICS or IMS definitions? Check the triggering attributes of the transmission queue.

Does the problem occur at specific times of the day?

If the problem occurs at specific times of day, it could be that it is dependent on system loading. Typically, peak system loading is at mid-morning and mid-afternoon, and so these are the times when load-dependent problems are most likely to occur. (If your network extends across more than one time zone, peak system loading might seem to occur at some other time of day.)

If you think that your WebSphere MQ for z/OS system has a performance problem, refer to “Dealing with performance problems” on page 27.

Does the problem affect all users of the application?

If the problem only affects some users, is this because some users do not have the correct security authorization? See the WebSphere MQ for z/OS System Setup Guide for information about all aspects of security for WebSphere MQ for z/OS.

Does the problem occur with all z/OS, CICS, or IMS systems?

If the problem only occurs when you access a particular z/OS, IMS, or CICS system, consider what is different about this system. Also consider whether any changes have been made to the system that might affect the way it interacts with WebSphere MQ.

Is the problem intermittent?

An intermittent problem could be caused by failing to take into account the fact that processes can run independently of each other. For example, a program might issue an **MQGET** call, without specifying WAIT, before an earlier process has completed. You might also encounter this if your application tries to get a message from a queue while it is in syncpoint (that is, before it has been committed).

Have you applied any APARs or PTFs?

If an APAR or PTF has been applied to WebSphere MQ for z/OS, check that no error message was produced. If the installation was successful, check with the IBM support center for any APAR or PTF error.

If an APAR or PTF has been applied to any other product, consider the effect it might have on the way WebSphere MQ interfaces with it.

Ensure that you have followed any instructions in the APAR that affect your system. (For example, you might have to redefine a resource.)

Do you have a program error?

The examples that follow illustrate the most common causes of problems encountered while running WebSphere MQ programs. You should consider the possibility that the problem with your system could be caused by one of these errors.

- Programs issue MQSET to change queue attributes and fail to reset attributes of a queue. For example, setting a queue to NOTRIGGER.
- Making incorrect assumptions about the attributes of a queue. This could include assuming that queues can be opened with MQOPEN when they are MQOPEN-exclusive, and assuming that queues are not part of a cluster when they are.
- Trying to access queues and data without the correct security authorization.
- Linking a program with no stub, or with the wrong stub (for example, a TSO program with the CICS stub). This can cause either a long-running unit of work, or an X'0C4' or other abend.
- Passing incorrect or invalid parameters in an MQI call; if the wrong number of parameters are passed, no attempt can be made to complete the completion code and reason code fields, and the task is abended. (This is an X'0C4' abend.)

  This problem might occur if you attempt to run an application on an earlier version of MQSeries® than it was written for, where some of the MQI values are invalid.
- Failing to define the WebSphere MQ modules to z/OS correctly (this causes an X'0C4' abend in CSQYASCP).
- Failing to check return codes from MQI requests.

  This problem might occur if you attempt to run an application on a later version of WebSphere MQ than it was written for, where new return codes have been introduced that are not checked for.
- Failing to open objects with the correct options needed for later MQI calls, for example using the MQOPEN call to open a queue but not specifying the correct options to enable the queue for subsequent MQGET calls.
- Failing to initialize *MsgId* and *CorrelId* correctly.

  This is especially true for MQGET.
- Using incorrect addresses.
- Using storage before it has been initialized.
- Passing variables with incorrect lengths specified.
- Passing parameters in the wrong order.
- Failing to define the correct security profiles and classes to RACF.

This might stop the queue manager or prevent you from carrying out any productive work.

- Relying on default MQI options for a ported application.

  For example, z/OS defaults to MQGET and MQPUT in syncpoint. The distributed-platform default is out of syncpoint.

- Relying on default behavior at a normal or abnormal end of a portal application.

  On z/OS, a normal end does an implicit MQCMIT and an abnormal end does an implicit rollback.

What to do next

Perhaps the preliminary checks have enabled you to find the cause of the problem. If so, you should now be able to resolve it, possibly with the help of other books in the WebSphere MQ library, and in the libraries of other licensed programs.

If you have not yet found the cause, you must start to look at the problem in greater detail. Begin by running the tests described in "Examining the problem in greater depth" to try to determine what type of problem you have.

Examining the problem in greater depth

The purpose of this chapter is to help you identify the cause of your problem if the preliminary checks have not enabled you to solve it.

Before you read this chapter, you should have worked through "Preliminary checks" on page 2. You also need access to the console log, and to diagnostic tools.

When you have established that no changes have been made to your system, and that there are no problems with your application programs, choose the option from the list below that best describes the symptoms of your problem:

- "Have you obtained some incorrect output?"
- "Have you received an unexpected error message?" on page 10
- "Has there been an abend?" on page 10
- "Have you failed to receive a response from an MQSC command?" on page 11
- "Is there a problem with the WebSphere MQ queues?" on page 13
- "Is your application or WebSphere MQ for z/OS running slowly?" on page 16
- "Has your application or WebSphere MQ for z/OS stopped processing work?" on page 17

If none of these symptoms describe your problem, consider whether it might have been caused by another component of your system.

Have you obtained some incorrect output?

If you have obtained what you believe to be some incorrect output, consider the following:

Classifying incorrect output

"Incorrect output" might be regarded as any sort of output that you were not expecting. However, use this term with care in the context of problem determination because it might be a secondary effect of some other type of

error. For example, looping could be occurring if you get any sort of repetitive output, even though that output is what you expected.

Error messages
WebSphere MQ also responds to many errors it detects by sending error messages. You might regard these messages as "incorrect output", but they are only symptoms of another type of problem. If you have received an error message from WebSphere MQ that you were not expecting, refer to "Have you received an unexpected error message?."

Unexpected messages
If your application has not received a message that it was expecting, has received a message containing unexpected or corrupted information, or has received a message that it was not expecting (for example, one that was destined for a different application), refer to "Dealing with incorrect output" on page 33.

Have you received an unexpected error message?

If your application has received an unexpected error message, consider whether the error message has originated from WebSphere MQ or from another program.

WebSphere MQ error messages

WebSphere MQ for z/OS error messages are prefixed with the letters CSQ.

If you get an unexpected WebSphere MQ error message (for example, in the console log, or the CICS log), look in the WebSphere MQ for z/OS Messages and Codes manual for an explanation.

The WebSphere MQ for z/OS Messages and Codes manual might give you enough information to resolve the problem quickly, or it might redirect you to another manual for further guidance. If you cannot deal with the message, you might have to contact the IBM support center for help.

Non-WebSphere MQ error messages

If you get an error message from another IBM program, or from the operating system, look in the messages and codes manual from the appropriate library for an explanation of what it means.

In a queue-sharing environment, look for the following error messages:

- XES (prefixed with the letters IXL)
- DB2 (prefixed with the letters DSN)
- RRS (prefixed with the letters ATR)

See "Message keyword" on page 85 for a list of the most common message prefixes.

Unexpected return codes
If your application has received an unexpected return code from WebSphere MQ, see the WebSphere MQ Application Programming Reference manual for information about how your application should deal with WebSphere MQ return codes.

Has there been an abend?

If your application has stopped running, this could be caused by an abnormal termination (abend).

You are notified of an abend in one of the following places, depending on what type of application you are using:

Batch Your listing shows the abend.

CICS You see a CICS transaction abend message. If your task is a terminal task, this message appears on your screen. If your task is not attached to a terminal, the message appears on the CICS CSMT log.

IMS In all cases, you see a message at the IMS master terminal and in the listing of the dependent region involved. If an IMS transaction that had been entered from a terminal was being processed, an error message is also sent to that terminal.

TSO You might see a TSO message with a return code on your screen. (This depends on the way your system is set up, and the type of error.)

Common causes of abends

Abends can be caused by the user ending the task being performed before it terminates normally (for example, purging a CICS transaction), or by an error in an application program.

Address space dumps and transaction dumps

For some abends, an address space dump is produced. For CICS transactions, a transaction dump showing the storage areas of interest to the transaction is provided.

- If an application passes some data, the address of which is no longer valid, a dump is sometimes produced in the user's address space.

  Note: For a batch dump, the dump is formatted and written to SYSUDUMP. For information about SYSUDUMPs, refer to "SYSUDUMP information" on page 62. For CICS, a system dump is written to the SYS1.DUMP data sets, as well as a transaction dump being taken.
- If a problem with WebSphere MQ for z/OS itself causes an abend, an abend code of X'5C6' or X'6C6' is returned, along with an abend reason code. This uniquely describes the cause of the problem. Refer to "WebSphere MQ for z/OS abends" on page 42 for information about the abend codes, and see the WebSphere MQ for z/OS Messages and Codes manual for an explanation of the reason code.

Abnormal program termination

If your program has terminated abnormally, refer to "Dealing with program abends" on page 21.

If your system has terminated abnormally, and you want to analyze the dump produced, refer to "WebSphere MQ dumps" on page 48. This chapter tells you how to format the dump, and how to interpret the data contained in it.

Have you failed to receive a response from an MQSC command?

If you have issued an MQSC command from an application (and not from a z/OS console), but you have not received a response, consider the following questions:

Is the command server running?

Check that the command server is running, as follows:

1. Use the DISPLAY CMDSERV command at the z/OS console to display the status of the command server.
2. If the command server is not running, start it using the START CMDSERV command.
3. If the command server is running, issue the DISPLAY QUEUE command, using the name of the system-command input queue and the CURDEPTH and MAXDEPTH attributes to define the data displayed.

   If these values show that the queue is full, and the command server has been started, this indicates that messages are not being read from the queue.
4. Try stopping the command server and then restarting it, responding to any error messages that are produced.
5. Issue the display command again to see if it is working now.

Has a reply been sent to the dead-letter queue?

Use the DISPLAY QMGR DEADQ command to find out the name of the system dead-letter queue (if you do not know what it is).

Use this name in the DISPLAY QUEUE command with the CURDEPTH attribute to see if there are any messages on the queue.

The dead-letter queue message header (dead-letter header structure) contains a reason or feedback code describing the problem. (See the WebSphere MQ Application Programming Reference manual for information about the dead-letter header structure.)

Are the queues enabled for PUTs and GETs?

Use the DISPLAY QUEUE command from the console to check, for example, DISPLAY QUEUE(SYSTEM.COMMAND.INPUT) PUT GET.

Is the `WaitInterval` parameter set to a sufficiently long time?

If your MQGET call has timed out, you will see a completion code of 2 and a reason code of 2033 (MQRC_NO_MSG_AVAILABLE). (See the WebSphere MQ Application Programming Guide for information about the *`WaitInterval`* parameter, and completion and reason codes from MQGET.)

Is a syncpoint required?

If you are using your own application program to put commands onto the system-command input queue, consider whether you need to take a syncpoint.

You need to take a syncpoint after putting messages to a queue, and before attempting to receive reply messages, or use MQPMO_NO_SYNCPOINT when putting them. Unless you have specifically excluded your request message from syncpoint, you need to take a syncpoint before attempting to receive reply messages.

Are the `MaxDepth` and `MaxMsgL` parameters of your queues set sufficiently high?

See the WebSphere MQ for z/OS Concepts and Planning Guide for information about defining the system-command input queue and the reply-to queue.

Are you using the *`CorrelId`* and *`MsgId`* parameters correctly?

You need to identify the queue and then display the CURDEPTH. Use the DISPLAY QUEUE command from the console, for example, DISPLAY

QUEUE (MY.REPLY.QUEUE) CURDEPTH, to see if there are messages on the reply-to queue that you have not received.

Set the values of *MsgId* and *CorrelId* in your application to ensure that you receive all messages from the queue. See the WebSphere MQ Application Programming Guide for information about using these fields.

The following questions are applicable if you have issued an MQSC command from either a z/OS console (or its equivalent), or an application, but have not received a response:

Is the queue manager still running, or did your command cause an abend?
Look for error messages indicating an abend, and if one occurred, refer to "WebSphere MQ dumps" on page 48

Were any error messages issued?
Check to see if any error messages were issued that might indicate the nature of the error.

See the WebSphere MQ for z/OS Concepts and Planning Guide for information about the different methods you can use to enter MQSC commands.

Is there a problem with the WebSphere MQ queues?

If you suspect that there is a problem affecting the queues on your subsystem, use the operations and control panels to display the system-command input queue.

If the system responds
This proves that at least one queue is working, so follow the procedure shown in "Are some of your queues working?."

If the system does not respond

The problem might be with the whole subsystem. In this instance, try stopping and restarting the queue manager, responding to any error messages that are produced.

Check for any messages on the console needing action, and resolve any that might affect WebSphere MQ, such as a request to mount a tape for an archive log. See if other subsystems or CICS regions are affected.

Use the DISPLAY QMGR COMMANDQ command to identify the name of the system command input queue.

If the problem still occurs after restart
Contact your IBM support center for help (see "Reporting a problem to the IBM software support group" on page 91).

Are some of your queues working?

If you suspect that the problem occurs with only a subset of queues, select the name of a local queue that you think is having problems and perform the following procedures:

Display queue information
Use the DISPLAY QUEUE and DISPLAY QSTATUS commands to display information about the queue.

Is the queue being processed?

- If CURDEPTH is at MAXDEPTH, this might indicate that the queue is not being processed. Check that all applications that use the queue are

running normally (for example, check that transactions in your CICS system are running or that applications started in response to Queue Depth High events are running).

- Issue DISPLAY QSTATUS(xx) IPPROCS to see if the queue is open for input. If not, start the application.
- If CURDEPTH is not at MAXDEPTH, check the following queue attributes to ensure that they are correct:
  - If triggering is being used:
    - Is the trigger monitor running?
    - Is the trigger depth too big?
    - Is the process name correct?
    - Have **all** the trigger conditions been met?

      Issue DISPLAY QSTATUS(xx) IPPROCS to see if an application has the same queue open for input. In some triggering scenarios, a trigger message is not produced if the queue is open for input. Stop the application to cause the triggering processing to be invoked.
  - Can the queue be shared? If not, another application (batch, IMS, or CICS) might already have it open for input.
  - Is the queue enabled appropriately for GET and PUT?

Do you have a long-running unit of work?

If CURDEPTH is not zero, but when you attempt to MQGET a message the queue manager replies that there is no message available, issue either DIS QSTATUS(xx) TYPE(HANDLE) to show you information about applications that have the queue open, or issue DIS CONN(xx) to give you more information about an application that is connected to the queue.

How many tasks are accessing the queues?

Issue DISPLAY QSTATUS(xx) OPPROCS IPPROCS to see how many tasks are putting messages on to, and getting messages from the queue. In a queue-sharing environment, check OPPROCS and IPPROCS on each queue manager. Alternatively, use the CMDSCOPE attribute to check all the queue managers. If there are no application processes getting messages from the queue, determine why this is (for example, because the applications need to be started, a connection has been disrupted, or because the MQOPEN call has failed for some reason).

Is this queue a shared queue? Does the problem affect only shared queues?

Check that there is not a problem with the sysplex elements that support shared queues. For example, check that there is not a problem with the WebSphere MQ-managed Coupling Facility list structure.

Use D XCF, STRUCTURE, STRNAME=ALL to check that the Coupling Facility structures are accessible.

Use D RRS to check that RRS is active.

Is this queue part of a cluster?

Check to see if the queue is part of a cluster (from the CLUSTER or CLUSNL attribute). If it is, verify that the queue manager that hosts the queue is still active in the cluster.

If you cannot solve the problem

Contact your IBM support center for help (see "Reporting a problem to the IBM software support group" on page 91).

Are the correct queues defined?

Check that the system-command input queue, the system-command reply model queue, and the reply-to queue are correctly defined, and that the MQOPEN calls were successful.

If you are using the system-command reply model queue, check that it was defined correctly.

If you are using clusters, you need to define the SYSTEM.CLUSTER.COMMAND.QUEUE to use commands relating to cluster processing.

Does the problem affect only remote or cluster queues?

If the problem affects only remote or cluster queues, check the following:

Are the remote queues being accessed?
Check that the programs that should be putting messages to the remote queues have run successfully (see "Dealing with incorrect output" on page 33).

Is the system link active?

Use APPC or TCP/IP commands as appropriate to check whether the link between the two systems is active.

Use PING or OPING for TCP/IP or D NET ID=xxxxx, E for APPC.

Is triggering working?
If you use triggering to start the distributed queuing process, check that the transmission queue has triggering set on and that the queue is get-enabled.

Is the channel or listener running?

If necessary, start the channel or the listener manually, or try stopping and restarting the channel. See the WebSphere MQ Intercommunication manual for information about how to do this.

Look for error messages on the startup of the channel initiator and listener. Use the WebSphere MQ for z/OS Messages and Codes manual and WebSphere MQ Intercommunication to determine the cause.

What is the channel status?
Check the channel status using the DISPLAY CHSTATUS (channelname) command.

Are your process and channel definitions correct?
Check your process definitions and your channel definitions.

See the WebSphere MQ Intercommunication manual for information about how to use distributed queuing, and for information about how to define channels.

Does the problem affect only shared queues?

If the problem affects only queue-sharing groups, use the VERIFY QSG function of the CSQ5PQSG utility. This command verifies that the DB2 setup is consistent in terms of the bitmap allocation fields, and object definition for the DB2 queue manager, structure, and shared queue objects, and reports details of any inconsistency that is discovered.

The following is an example of a VERIFY QSG report with errors:

```
CSQU501I   VERIFY QSG function requested
CSQU503I   QSG=SQ02, DB2 DSG=DSN710P5, DB2 ssid=DFP5
CSQU517I   XCF group CSQGSQ02 already defined
CSQU520I   Summary information for XCF group CSQGSQ02
CSQU522I   Member=MQ04, state=QUIESCED, system=MV4A
CSQU523I   User data=D4E5F4C15AD4D8F0F4404040C4C5....
CSQU522I   Member=MQ03, state=QUIESCED, system=MV4A
CSQU523I   User data=D4E5F4C15AD4D8F0F3404040C4C6....
CSQU526I   Connected to DB2 DF4A
CSQU572E   Usage map T01_ARRAY_QMGR and DB2 table CSQ.ADMIN_B_QMGR inconsistent
CSQU573E   QMGR MQ04 in table entry 1 not set in usage map
CSQU574E   QMGR 27 in usage map has no entry in table
CSQU572E   Usage map T01_ARRAY_STRUC and DB2 table CSQ.ADMIN_B_STRUCTURE inconsistent
CSQU575E   Structure APPL2 in table entry 4 not set in usage map
CSQU576E   Structure 55 in usage map has no entry in table
CSQU572E   Usage map T03_LH_ARRAY and DB2 table CSQ.OBJ_B_QUEUE inconsistent
CSQU577E   Queue MYSQ in table entry 13 not set in usage map for structure APPL1
CSQU576E   Queue 129 in usage map for structure APPL1 has no entry in table
CSQU528I    Disconnected from DB2 DF4A
CSQU148I    CSQ5PQSG Utility completed, return code=12
```

Is your application or WebSphere MQ for z/OS running slowly?

If your application is running slowly, this could indicate that it is in a loop, or waiting for a resource that is not available.

Is the problem worse at peak system load times?

This could also be caused by a performance problem. Perhaps it is because your system needs tuning, or because it is operating near the limits of its capacity. This type of problem is probably worst at peak system load times, typically at mid-morning and mid-afternoon. (If your network extends across more than one time zone, peak system load might seem to you to occur at some other time.)

Does the problem occur when the system is lightly loaded?

If you find that degrading performance is not dependent on system loading, but happens sometimes when the system is lightly loaded, a poorly designed application program is probably to blame. This could manifest itself as a problem that only occurs when specific queues are accessed.

Is WebSphere MQ for z/OS running slowly?

The following symptoms might indicate that WebSphere MQ for z/OS is running slowly:

- If your system is slow to respond to commands.
- If repeated displays of the queue depth indicate that the queue is being processed slowly for an application with which you would expect a large amount of queue activity.

You can find guidance on dealing with waits and loops in “Dealing with waits and loops” on page 22, and on dealing with performance problems in “Dealing with performance problems” on page 27.

Has your application or WebSphere MQ for z/OS stopped processing work?

There are several reasons why your system would unexpectedly stop processing work. These include:

Queue manager problems
The queue manager could be shutting down.

Application problems
An application programming error could mean that the program branches away from its normal processing, or the application could get in a loop. There could also have been an application abend.

WebSphere MQ problems
Your queues could have become disabled for MQPUT or MQGET calls, the dead-letter queue could be full, or WebSphere MQ for z/OS could be in a wait state, or a loop.

z/OS and other system problems
z/OS could be in a wait state, or CICS or IMS could be in a wait state or a loop. There could be problems at the system or sysplex level that are affecting the queue manager or the channel initiator. For example, excessive paging. It could also indicate DASD problems, or higher priority tasks using a lot of CPU.

DB2 and RRS problems
Check that DB2 and RRS are active.

In all cases, carry out the following checks to determine the cause of the problem:

Check for error messages

Issue the DISPLAY THREAD(*) command to check if the queue manager is running (see the WebSphere MQ Script (MQSC) Command Reference manual for information about the command). If the queue manager has stopped running, look for any message that might explain the situation. Messages appear on the z/OS console, or on your terminal if you are using the operations and control panels. Use the DISPLAY DQM command to see if the channel initiator is working, and the listeners are active. The z/OS command

```
DISPLAY R,L
```

lists messages with outstanding replies. Check to see whether any of these are relevant. In some circumstances, for example, when it has used all its active logs, WebSphere MQ for z/OS waits for operator intervention.

No error messages issued

If no error messages have been issued, perform the following procedure to determine what is causing the problem:

1. Issue the z/OS commands

   ```
   DISPLAY A,xxxxMSTR
   DISPLAY A,xxxxCHIN
   ```

   (where xxxx is the WebSphere MQ for z/OS subsystem name). If you receive a message telling you that the queue manager or channel initiator has not been found, this indicates that the subsystem has terminated. This could be caused by an abend or by operator shutdown of the system.

2. If the subsystem is running, you will receive message IEE105I. This message includes the *CT=nnnn* field, which contains information about the processor time being used by the subsystem. Note the value of this field, and reissue the command.
   - If the *CT=* value has not changed, this indicates that the subsystem is not using any processor time. This could indicate that the subsystem is in a wait state (or that it has no work to do). If you can issue a command like DISPLAY DQM and you get output back, this indicates there is no work to do rather than a hang condition.
   - If the *CT=* value has changed dramatically, and continues to do so over repeated displays, this could indicate that the subsystem is very busy or possibly in a loop.
   - If the reply indicates that the subsystem is now not found, this indicates that it was in the process of terminating when the first command was issued. If a dump is being taken, the subsystem might take a while to terminate. A message is produced at the console before terminating.

     To check that the channel initiator is working, issue the DISPLAY DQM command. If the response does not show the channel initiator working this could be because it is getting insufficient resources (like the CPU). In this case, use the z/OS monitoring tools, such as RMF™, to determine if there is a resource problem. If it is not, restart the channel initiator.

Has the queue manager or channel initiator terminated abnormally?

Look for any messages saying that the queue manager or channel initiator address space has abnormally terminated. If you get a message for which the system action is to terminate WebSphere MQ, find out if a system dump was produced, and turn to "WebSphere MQ dumps" on page 48.

WebSphere MQ for z/OS might still be running

Consider also that WebSphere MQ for z/OS might still be running, but only slowly. If it *is* running slowly, you probably have a performance problem. If so, read "Is your application or WebSphere MQ for z/OS running slowly?" on page 16 to confirm this before going on. Refer to "Dealing with performance problems" on page 27 for advice about what to do next.

Where to look next

If you have been successful in identifying the cause of your problem, use Table 1 to determine your next course of action.

Table 1. Where to look next

Type of problem	Where to look next
Program abend	"Dealing with program abends" on page 21
System abend	"WebSphere MQ dumps" on page 48
Wait	"Dealing with waits and loops" on page 22
Loop	"Dealing with waits and loops" on page 22
Performance problem	"Dealing with performance problems" on page 27
Incorrect output	"Dealing with incorrect output" on page 33
Unexpected message	WebSphere MQ for z/OS Messages and Codes manual

Table 1. Where to look next (continued)

Type of problem	Where to look next
Application error	WebSphere MQ Application Programming Guide and WebSphere MQ Application Programming Reference manual

If you have decided that the problem is with the WebSphere MQ code itself, and that you should refer it to the IBM support center, you can get advice on dealing with the support center in "Reporting a problem to the IBM software support group" on page 91.

Chapter 2. Dealing with the problem

Dealing with program abends

Program abends can be caused by applications failing to check, and respond to, reason codes from WebSphere MQ. For example, if a message has not actually been received, using fields that would have been set up in the message for calculation could cause X'0C4' or X'0C7' abends (ASRA abends in CICS).

The following pieces of information indicate a program abend:

- Error messages from WebSphere MQ in the console log
- CICS error messages
- CICS transaction dumps
- IMS region dumps
- IMS messages on user or master terminal
- Program dump information in batch or TSO output
- Abend messages in batch job output
- Abend messages on the TSO screen

If you have an abend code, see one of the following manuals for an explanation of the cause of the abend:

- For WebSphere MQ for z/OS abends (abend codes X'5C6' and X'6C6'), the WebSphere MQ for z/OS Messages and Codes manual
- For batch abends, the *MVS System Codes* manual
- For CICS abends, the *CICS Messages and Codes* manual
- For IMS abends, the *IMS/ESA® Messages and Codes* manual
- For DB2 abends, the *DB2 Messages and Codes* manual
- For RRS abends, the *MVS System Messages* manual
- For XES abends, the *MVS System Messages* manual

Batch abends

Batch abends cause an error message containing information about the contents of registers to appear in the syslog. TSO abends cause an error message containing similar information to be produced on the TSO screen. A SYSUDUMP is taken if there is a SYSUDUMP DD statement for the step (refer to "WebSphere MQ dumps" on page 48 for information about SYSUDUMPs).

CICS transaction abends

CICS transaction abends are recorded in the CICS CSMT log, and a message is produced at the terminal (if there is one). A CICS AICA abend indicates a possible loop. Refer to "Dealing with loops" on page 25 for more information. If you have a CICS abend, using CEDF and the CICS trace might help you to find the cause of the problem. See the *CICS Problem Determination Guide* for more information.

IMS transaction abends

IMS transaction abends are recorded on the IMS master terminal, and an error message is produced at the terminal (if there is one). If you have an IMS abend, see the *IMS/ESA Diagnosis Guide and Reference* manual.

Dealing with waits and loops

To perform the tests shown in this chapter, you need access to the z/OS console, and to be able to issue operator commands.

Waits and loops are characterized by unresponsiveness. However, it can be quite difficult to distinguish between waits, loops, and poor performance.

Any of the following symptoms could be caused by a wait or a loop, or by a badly tuned or overloaded system:

- An application that appears to have stopped running (if WebSphere MQ for z/OS is still responsive, this is probably an application problem)
- An MQSC command that does not produce a response
- Excessive use of processor time (CPU)

Distinguishing between waits and loops

Because waits and loops can be difficult to distinguish, in some cases you need to carry out a detailed investigation before deciding which classification is right for your problem.

This section gives you guidance about choosing the best classification, and advice on what to do when you have decided on a classification.

Waits

For the purpose of problem determination, a wait state is regarded as the state in which the execution of a task has been suspended. That is, the task has started to run, but has been suspended without completing, and has subsequently been unable to resume.

A problem identified as a wait in your system could be caused by any of the following:

- A wait on an MQI call
- A wait on a CICS or IMS call
- A wait for another resource (for example, file I/O)
- An ECB wait
- The CICS or IMS region waiting
- TSO waiting
- WebSphere MQ for z/OS waiting for work
- An apparent wait, caused by a loop
- Your task is not being dispatched by CICS or MVS™ due to higher priority work
- DB2 or RRS are inactive

Loops

A loop is the repeated execution of some code. If you have not planned the loop, or if you have designed it into your application but it does not terminate for some reason, you get a set of symptoms that vary depending on what the code is doing, and how any interfacing components and products react to it. In some cases, at first, a loop might be diagnosed as a wait or performance problem, because the looping task competes for system resources with other tasks that are not involved in the loop. However, a loop consumes resources but a wait does not.

An apparent loop problem in your system could be caused by any of the following:

- An application doing a lot more processing than usual and therefore taking much longer to complete
- A loop in application logic
- A loop with MQI calls
- A loop with CICS or IMS calls
- A loop in CICS or IMS code
- A loop in WebSphere MQ for z/OS

Symptoms of waits and loops

Any of the following symptoms could be caused by a wait, a loop, or by a badly tuned or overloaded system:

- Timeouts on MQGET WAITs
- Batch jobs suspended
- TSO session suspended
- CICS task suspended
- Transactions not being started because of resource constraints, for example CICS MAX task
- Queues becoming full, and not being processed
- System commands not accepted, or producing no response

Dealing with waits

When investigating what appears to be a problem with tasks or subsystems waiting, it is necessary to take into account the environment in which the task or subsystem is running.

It might be that your z/OS system is generally under stress. In this case, there will be many symptoms. If there is not enough real storage, jobs will experience waits at paging interrupts or swap-outs. Input/output (I/O) contention or high channel usage can also cause waits.

You can use standard monitoring tools, such as *Resource Monitoring Facility* (RMF) to diagnose such problems. Use normal z/OS tuning techniques to resolve them.

Is a batch or TSO program waiting?

Consider the following points:

Your program might be waiting on another resource
For example, a VSAM control interval (CI) that another program is holding for update.

Your program might be waiting for a message that has not yet arrived

This might be normal behavior if, for example, it is a server program that constantly monitors a queue.

Alternatively, your program might be waiting for a message that has arrived, but has not yet been committed.

Issue the DIS CONN(*) TYPE(HANDLE) command and examine the queues in use by your program.

If you suspect that your program has issued an MQI call that did not involve an MQGET WAIT, and control has not returned from WebSphere MQ, take an SVC dump of both the batch or TSO job, and the WebSphere MQ subsystem before cancelling the batch or TSO program.

Also consider that the wait state might be the result of a problem with another program, such as an abnormal termination (see "Messages do not appear when expected" on page 33), or in WebSphere MQ itself (see "Is WebSphere MQ for z/OS waiting?" on page 25). Refer to "WebSphere MQ dumps" on page 48 (specifically Figure 2 on page 50) for information about obtaining a dump.

If the problem persists, refer to Chapter 4, "Finding solutions to similar problems," on page 75 and Chapter 5, "Working with IBM to solve your problem," on page 91 for information on reporting the problem to IBM.

Is a CICS transaction waiting?

Consider the following points:

CICS could be under stress
This could indicate that the maximum number of tasks allowed (MAXTASK) has been reached, or a short on storage (SOS) condition exists. Check the console log for messages that might explain this (for example, SOS messages), or see the *CICS Problem Determination Guide*.

The transaction could be waiting for another resource
For example, this could be file I/O. You can use CEMT INQ TASK to see what the task is waiting for. If the resource type is MQSERIES your transaction is waiting on WebSphere MQ (either in an **MQGET** WAIT or a task switch). Otherwise see the *CICS Problem Determination Guide* to determine the reason for the wait.

The transaction could be waiting for WebSphere MQ for z/OS
This could be normal, for example, if your program is a server program that waits for messages to arrive on a queue. Otherwise it might be the result of a transaction abend, for example (see "Messages do not appear when expected" on page 33). If this is the case, the abend is reported in the CSMT log.

The transaction could be waiting for a remote message
If you are using distributed queuing, the program might be waiting for a message that has not yet been delivered from a remote system (for further information, refer to "Problems with missing messages when using distributed queuing" on page 35).

If you suspect that your program has issued an MQI call that did not involve an **MQGET** WAIT (that is, it is in a task switch), and control has not returned from WebSphere MQ, take an SVC dump of both the CICS region, and the WebSphere MQ subsystem before cancelling the CICS transaction. Refer to "Is WebSphere MQ for z/OS waiting?" for information about waits. Refer to "WebSphere MQ dumps" on page 48 (specifically Figure 2 on page 50) for information about obtaining a dump.

If the problem persists, refer to Chapter 4, "Finding solutions to similar problems," on page 75 and Chapter 5, "Working with IBM to solve your problem," on page 91 for information on reporting the problem to IBM.

Is DB2 waiting?

If your investigations indicate that DB2 is waiting, check the following:

1. Use the DB2 -DISPLAY THREAD(*) command to determine if any activity is taking place between the queue manager and the DB2 subsystem.
2. Try and determine whether any waits are local to the queue manager subsystems or are across the DB2 subsystems.

Is RRS active?

- Use the D RRS command to determine if RRS is active.

Is WebSphere MQ for z/OS waiting?

If your investigations indicate that WebSphere MQ itself is waiting, check the following:

1. Use the DISPLAY THREAD(*) command to check if anything is connected to WebSphere MQ.
2. Use SDSF DA, or the z/OS command `DISPLAY A,xxxxMSTR` to determine whether there is any CPU usage (as shown in "Has your application or WebSphere MQ for z/OS stopped processing work?" on page 17).
   - If WebSphere MQ is using some CPU, reconsider other reasons why WebSphere MQ might be waiting, or consider whether this is actually a performance problem.
   - If there is no CPU activity, check whether WebSphere MQ responds to commands. If you can get a response, reconsider other reasons why WebSphere MQ might be waiting.
   - If you cannot get a response, check the console log for messages that might explain the wait (for example, WebSphere MQ might have run out of active log data sets, and be waiting for an off-load).

If you are satisfied that WebSphere MQ has actually stalled, use the STOP QMGR command in both QUIESCE and FORCE mode to terminate any programs currently being executed.

If the STOP QMGR command fails to respond, cancel the queue manager with a dump, and restart. If the problem recurs, refer to Chapter 5, "Working with IBM to solve your problem," on page 91 for further guidance.

Dealing with loops

The following sections describe the various types of loop that you might encounter, and suggest some responses.

Is a batch application looping?

If you suspect that a batch or TSO application is looping, use the console to issue the z/OS command `DISPLAY JOBS,A` (for a batch application) or `DISPLAY TS,A` (for a TSO application). Note the CT values from the data displayed, and repeat the command.

If any task shows a significant increase in the CT value, it might be that the task is looping. You could also use SDSF DA, which shows you the percentage of CPU that each address space is using.

Is a batch job producing a large amount of output?

An example of this might be an application that browses a queue and prints the messages. If the browse operation has been started with BROWSE FIRST, and subsequent calls have not been reset to BROWSE NEXT, the application will browse and print the first message on the queue repeatedly.

You can use SDSF DA to look at the output of running jobs if you suspect that this is a problem.

Does a CICS region show a lot of CPU activity?

It might be that a CICS application is looping, or that the CICS region itself is in a loop. You might see AICA abends if a transaction goes into a tight (unyielding) loop.

If you suspect that CICS, or a CICS application is looping, see the *CICS Problem Determination Guide*.

Does an IMS region show a lot of CPU activity?

It might be that an IMS application is looping. If you suspect this, see the *IMS Diagnosis Guide and Reference* manual.

Is the queue manager showing a lot of CPU activity?

Try to enter an MQSC DISPLAY command from the console. If you get no response, it is possible that the queue manager is looping. Follow the procedure shown in "Has your application or WebSphere MQ for z/OS stopped processing work?" on page 17 to display information about the processor time being used by the queue manager. If this indicates that the queue manager is in a loop, take a memory dump, cancel the queue manager and restart.

If the problem persists, refer to Chapter 4, "Finding solutions to similar problems," on page 75 and Chapter 5, "Working with IBM to solve your problem," on page 91 for information on reporting the problem to IBM.

Is a queue, page set, or Coupling Facility structure filling up unexpectedly?

This could indicate that an application is looping, and putting messages on to a queue. (This could be a batch, CICS, or TSO application.)

Identifying a looping application

In a busy system, it might be difficult to identify which application is causing the problem. If you keep a cross-reference of applications to

queues, terminate any programs or transactions that could be putting messages on to the queue, and investigate these programs or transactions before using them again. (The most likely culprits will be new, or changed applications; check your change log to identify them.)

Try issuing a DISPLAY QSTATUS command on the queue. This will give you information about the queue that might help to identify which application is looping.

Incorrect triggering definitions
It might be that a getting application has not been triggered due to incorrect object definitions, for example, the queue might be set to NOTRIGGER.

Distributed queuing
Using distributed queuing, a symptom of this problem could be a message in the receiving system indicating that **MQPUT** calls to the dead-letter queue are failing. This could be because the dead-letter queue has also filled up. The dead-letter queue message header (dead-letter header structure) contains a reason or feedback code explaining why the message could not be put on to the target queue. See the WebSphere MQ Application Programming Reference manual for information about the dead-letter header structure.

Allocation of queues to page sets
If a particular page set frequently fills up, there could be a problem with the allocation of queues to page sets. Refer to "Distribution of queues on page sets" on page 29 for more information.

Shared queues
Is the Coupling Facility structure full? The z/OS command DISPLAY CF displays information about Coupling Facility storage including the total amount, the total in use, and the total free control and noncontrol storage. The RMF Coupling Facility Usage Summary Report provides a more permanent copy of this information.

Are a task, and WebSphere MQ for z/OS, showing a lot of CPU activity?

In this case, a task might be looping on MQI calls (for example, browsing the same message repeatedly).

Dealing with performance problems

Performance problems are characterized by the following:

- Poor response times in online transactions
- Batch jobs taking a long time to complete
- The transmission of messages is slow

They can be caused by many factors, from a lack of resource in the z/OS system as a whole, to poor application design. This chapter presents problems and suggested solutions, starting with problems that are relatively simple to diagnose, such as DASD contention, through problems with specific subsystems, such as WebSphere MQ and CICS or IMS, and ending with the more subtle problems of application design, which might take more detecting.

Remote queuing problems can be due to network congestion and other network problems. They can also be caused by problems at the remote queue manager.

z/OS system considerations

You might already be aware that your z/OS system is under stress because these problems impact many subsystems and applications.

You can use the standard monitoring tools such as Resource Monitoring Facility (RMF) to monitor and diagnose these problems. They might include the following:

- Constraints on storage (paging)
- Constraints on CPU cycles
- Constraints on DASD
- Channel path usage

Use normal z/OS tuning techniques to resolve these problems.

WebSphere MQ for z/OS considerations

There are a number of decisions to be made when customizing WebSphere MQ for z/OS that can affect the way your systems perform. These include the following:

- The size and placement of data sets
- The allocation of buffers
- The distribution of queues among page sets, and Coupling Facility structures
- The number of tasks that you allow to access the queue manager at any one time

Log buffer pools

Insufficient log buffers can cause applications to wait until a log buffer is available, which can affect WebSphere MQ performance. RMF reports might show heavy I/O to volumes that hold log data sets.

There are three parameters you can use to tune log buffers. The most important is OUTBUFF. If the log manager statistic QJSTWTB is greater than 0, increase the size of the log buffer. This controls the number of buffers to be filled before they are written to the active log data sets (in the range 1 to 256). Commits and out-of-syncpoint processing of persistent messages cause log buffers to be written out to the log. As a result this parameter might have little effect except when processing large messages, and the number of commits or out of syncpoint messages is low. These parameters are specified in the CSQ6LOGP macro (see the WebSphere MQ for z/OS System Setup Guide for details), and the significant ones are:

OUTBUFF
This controls the size of the output buffer (in the range 40 KB through 4000 KB).

WRTHRSH
This controls the number of buffers to be filled before they are written to the active log data sets (in the range 1 through 256).

You should also be aware of the LOGLOAD parameter of the CSQ6SYSP macro. This specifies the number of log records that are written between checkpoint records. The range is 200 through 16 000 000 but a typical value for a large system

is 50 000. If a value is too small you will receive frequent checkpoints, which consume CPU and can cause additional disk I/O.

Buffer pool size

There is a buffer pool associated with each page set. You can specify the number of buffers in the buffer pool using the DEFINE BUFFPOOL command. See the WebSphere MQ for z/OS System Setup Guide manual for more information.

Incorrect specification of buffer pool size can adversely impact WebSphere MQ performance. The smaller the buffer pool, the more frequently physical I/O is required. RMF might show heavy I/O to volumes that hold page sets. For buffer pool with only short-lived messages the buffer manager statistics QPSTSLA, QPSTSOS, and QPSTRIO should usually be zero. For other buffer pools, QPSTSOS and QPSTSTLA should be zero.

Distribution of data sets on available DASD

The distribution of page data sets on DASD can have a significant impact on the performance of WebSphere MQ.

Place log data sets on low usage volumes with log *n* and log *n+1* on different volumes. Ensure that dual logs are placed on DASD on different control units and that the volumes are not on the same physical disk.

Distribution of queues on page sets

The distribution of queues on page sets can have an impact on performance. This can be indicated by poor response times experienced by transactions using specific queues that reside on heavily used page sets. RMF reports might show heavy I/O to volumes containing the affected page sets.

You can assign queues to specific page sets by defining storage class (STGCLASS) objects specifying a particular page set, and then defining the STGCLASS parameter in the queue definition. It is a good idea to define heavily-used queues on different page sets in this way.

Distribution of queues on Coupling Facility structures

The distribution of queues on Coupling Facility structures can have an impact on performance.

A queue-sharing group can connect to up to 64 Coupling Facility structures, one of which must be the administration structure. You can use the remaining 63 Coupling Facility structures for WebSphere MQ data with each structure holding up to 512 queues. If you need more than one Coupling Facility structure, separate the queues across several structures based on the queues' function.

There are some steps you can take to maximize efficiency:

- Delete any Coupling Facility structures you no longer require.
- Place all the queues used by an application on the same Coupling Facility to make application processing efficient.
- If work is particularly performance sensitive, choose a faster Coupling Facility structure.

Consider that if you lose a Coupling Facility structure, you lose any non-persistent messages stored in it. This can cause consistency problems if queues are spread across various Coupling Facility structures. To use persistent messages, you must define the Coupling Facility structures with at least CFLEVEL(3) and RECOVER(YES).

Limitation of concurrent threads

The number of tasks accessing the queue manager could also have an impact on performance, particularly if there are other constraints, such as storage, or there are a large number of tasks accessing a few queues. Symptoms of this could be heavy I/O against one or more page sets, or poor response times from tasks known to access the same queues.

There are two parameters specified in the CSQ6SYSP macro that control the number of threads (see the WebSphere MQ for z/OS System Setup Guide for details). These are:

IDFORE
Specifies the number of TSO sessions that can connect to the queue manager concurrently.

IDBACK
Specifies the number of batch tasks (excluding TSO tasks) that can connect to the queue manager.

In a CICS environment, you can use CICS MAXTASK to limit concurrent access.

Using the WebSphere MQ trace for administration

Although you need to use specific traces on occasion, using the trace facility has a negative impact on the performance of your systems.

Consider what destination you want your trace information sent to. Using the internal trace table saves I/O, but it is not large enough for traces that produce a lot of data.

The statistics trace gathers information at intervals. The intervals are controlled by the STATIME parameter of the CSQ6SYSP macro, described in the WebSphere MQ for z/OS System Setup Guide. An accounting trace record is produced when the task or channel ends. This could be after many days.

You can limit traces by class, resource manager identifier (RMID), and instrumentation facility identifier (IFCID) to reduce the volume of data collected. See the WebSphere MQ Script (MQSC) Command Reference manual for more information.

CICS constraints

Performance of WebSphere MQ tasks can be affected by CICS constraints. For example, your system might have reached MAXTASK, forcing transactions to wait, or the CICS system might be short on storage. For example, CICS might not be scheduling transactions because the number of concurrent tasks has been reached, or CICS has detected a resource problem. If you suspect that CICS is causing your performance problems (for example because batch and TSO jobs run successfully, but your CICS tasks time out, or have poor response times), see the *CICS Problem Determination Guide* and the *CICS Performance Guide*.

Note: CICS I/O to transient data extrapartition data sets uses the z/OS RESERVE command. This could impact I/O to other data sets on the same volume.

Application design considerations

There are a number of ways in which poor program design can affect performance. These can be difficult to detect because the program can appear to perform well, while impacting the performance of other tasks. Several problems specific to programs making MQI calls are discussed in the following sections.

For more information about application design, see the WebSphere MQ Application Programming Guide.

Effect of message length

Although WebSphere MQ for z/OS allows messages to hold up to 100 MB of data, the amount of data in a message affects the performance of the application that processes the message. To achieve the best performance from your application, send only the essential data in a message; for example, in a request to debit a bank account, the only information that might need to be passed from the client to the server application is the account number and the amount of the debit.

Effect of message persistence

Persistent messages are logged. Logging messages reduces the performance of your application, so you should use persistent messages for essential data only. If the data in a message can be discarded if the queue manager stops or fails, use a nonpersistent message.

Data for persistent messages is written to log buffers. These buffers are written to the log data sets when:

- A commit occurs
- A message is got or put out of syncpoint
- WRTHRSH buffers are filled

If an application processes many messages in one unit of work this causes less input/output than if the messages were processed one for each unit of work, or out of syncpoint.

Searching for a particular message

The **MQGET** call usually retrieves the first message from a queue. If you use the message and correlation identifiers (*MsgId* and *CorrelId*) in the message descriptor to specify a particular message, the queue manager has to search the queue until it finds that message. Using the **MQGET** call in this way affects the performance of your application because, to find a particular message, WebSphere MQ might have to scan the entire queue.

You can use the *IndexType* queue attribute to specify that you want the queue manager to maintain an index that can be used to increase the speed of **MQGET** operations on the queue. However, there is a small performance overhead for maintaining an index, so only generate one if you know that you will use it. You can choose to build an index of message identifiers or of correlation identifiers, or you can choose not to build an index for queues where messages are retrieved sequentially. Try to have many different key values, not many with the same value. For example Balance1, Balance2, and Balance3, not three with Balance. For shared

queues, you must have the right *IndexType* as described in the WebSphere MQ Application Programming Reference manual.

To avoid affecting queue manager restart time by using indexed queues, use the QINDXBLD(NOWAIT) parameter in the CSQ6SYSP macro. This allows queue manager restart to complete without waiting for queue index building to complete.

For a full description of the *IndexType* attribute, see the WebSphere MQ Application Programming Reference manual.

Queues that contain messages of different lengths

Get a message, using a buffer size matching the expected size of the message. If you receive the return code indicating that the message is too long, get a bigger buffer. When the get fails in this way, the data length returned is the size of the unconverted message data. If you specify MQGMO_CONVERT on the **MQGET** call, and the data expands during conversion, it still might not fit in the buffer, in which case you need to further increase the size of the buffer.

If you issue the **MQGET** with a buffer length of zero, it returns the size of the message and the application can get a buffer of this size and reissue the get. If you have multiple applications processing the queue, another application might have already processed the message when the original application reissued the get. If you occasionally have very large messages, you might get a large buffer just for these messages, and release it after the message has been processed. This should help reduce virtual storage problems if all applications have very large buffers.

If your application cannot use messages of a fixed length, another solution to this problem is to use the **MQINQ** call to find the maximum size of messages that the queue can accept, then use this value in your **MQGET** call. The maximum size of messages for a queue is stored in the *MaxMsgL* attribute of the queue. This method could use large amounts of storage, however, because the value of this queue attribute could be as high as 100 MB, the maximum allowed by WebSphere MQ for z/OS.

Note: You can lower the *MaxMsgL* parameter after large messages have been put to the queue. For example you can put a 100 MB message, then set *MaxMsgL* to 50 bytes. This means it is still possible to get bigger messages than the application expected.

Frequency of syncpoints

Programs that issue a lot of **MQPUT** calls within syncpoint, without committing them, can cause performance problems. Affected queues can fill up with messages that are currently unusable, while other tasks might be waiting to get these messages. This has implications in terms of storage, and in terms of threads tied up with tasks that are attempting to get messages.

As a general rule if you have multiple applications processing a queue you usually get the best performance when you have either

- 100 short messages (less than 1 KB), or
- One message for larger messages (100 KB)

for each syncpoint. If there is only one application processing the queue, you must have more messages for each unit of work.

You can limit the number of messages that a task can get or put within a single unit of recovery with the MAXUMSGS queue manager attribute. See the ALTER QMGR command in the WebSphere MQ Script (MQSC) Command Reference for information about this attribute.

Advantages of the MQPUT1 call

Use the **MQPUT1** call only if you have a single message to put on a queue. If you want to put more than one message, use the **MQOPEN** call, followed by a series of **MQPUT** calls and a single **MQCLOSE** call.

Dealing with incorrect output

The term "incorrect output" can be interpreted in many different ways, and its meaning for the purpose of problem determination with this book is explained in "Examining the problem in greater depth" on page 9.

This chapter discusses the following problems that you could encounter with your system and classify as incorrect output:

- Application messages that do not appear when you are expecting them
- Application messages that contain the wrong information, or information that has been corrupted

Additional problems that you might encounter if your application uses distributed queues are also discussed.

Messages do not appear when expected

If messages do not appear on the queue when you are expecting them, check for the following:

Has the message been put onto the queue successfully?

Did WebSphere MQ issue a return and reason code for the MQPUT, for example:

- Has the queue been defined correctly, for example is MAXMSGL large enough? (reason code 2030).
- Can applications put messages on to the queue (is the queue enabled for MQPUT calls)? (reason code 2051).
- Is the queue already full? This could mean that an application could not put the required message on to the queue (reason code 2053).

Is the queue a shared queue?

- Have Coupling Facility structures been defined successfully in the CFRM policy data set? Messages held on shared queues are stored inside a Coupling Facility.
- Have you activated the CFRM policy?

Is the queue a cluster queue?

If it is, there might be multiple instances of the queue on different queue managers. This means the messages could be on a different queue manager.

- Did you want the message to go to a cluster queue?
- Is your application designed to work with cluster queues?

- Did the message get put to a different instance of the queue from that expected?

Check any cluster-workload exit programs to see that they are processing messages as intended.

Do your gets fail?

- Does the application need to take a syncpoint?

  If messages are being put or got within syncpoint, they are not available to other tasks until the unit of recovery has been committed.
- Is the time interval on the MQGET long enough?

  If you are using distributed processing, you should allow for reasonable network delays, or problems at the remote end.
- Was the message you are expecting defined as persistent?

  If not, and the queue manager has been restarted, the message will have been deleted. Shared queues are an exception because nonpersistent messages survive a queue manager restart.
- Are you waiting for a specific message that is identified by a message or correlation identifier (*MsgId* or *CorrelId*)?

  Check that you are waiting for a message with the correct *MsgId* or *CorrelId*. A successful MQGET call sets both these values to that of the message got, so you might need to reset these values to get another message successfully.

  Also check if you can get other messages from the queue.
- Can other applications get messages from the queue?

  If so, has another application already retrieved the message?

  If the queue is a shared queue, check that applications on other queue managers are not getting the messages.

If you cannot find anything wrong with the queue, and the queue manager itself is running, make the following checks on the process that you expected to put the message on to the queue:

- Did the application get started?

  If it should have been triggered, check that the correct trigger options were specified.
- Is a trigger monitor running?
- Was the trigger process defined correctly (both to WebSphere MQ for z/OS and CICS or IMS)?
- Did it complete correctly?

  Look for evidence of an abend, (for example, in the CICS log).
- Did the application commit its changes, or were they backed out?

  Look for messages in the CICS log indicating this.

If multiple transactions are serving the queue, they might occasionally conflict with one another. For example, one transaction might issue an MQGET call with a buffer length of zero to find out the length of the message, and then issue a specific MQGET call specifying the *MsgId* of that message. However, while this is happening, another transaction might have issued a successful MQGET call for that message, so the first application will receive a completion code of MQRC_NO_MSG_AVAILABLE. Applications that are expected to run in a multi-server environment must be designed to cope with this situation.

Have any of your systems suffered an outage? For example, if the message you were expecting should have been put on to the queue by a CICS application, and the CICS system went down, the message might be in doubt. This means that the queue manager does not know whether the message should be committed or backed out, and so has locked it until this is resolved when resynchronization takes place.

Note: The message is deleted after resynchronization if CICS decides to back it out.

Also consider that the message could have been received, but that your application failed to process it in some way. For example, did an error in the expected format of the message cause your program to reject it? If this is the case, refer to "Messages contain unexpected or corrupted information" on page 38.

Problems with missing messages when using distributed queuing

If your application uses distributed queuing, consider the following points:

Has distributed queuing been correctly installed on both the sending and receiving systems?

Ensure that the instructions about installing the distributed queue management facility in the WebSphere MQ for z/OS System Setup Guide have been followed correctly.

Are the links available between the two systems?

Check that both systems are available, and connected to WebSphere MQ for z/OS. Check that the LU 6.2 or TCP/IP connection between the two systems is active or check the connection definitions on any other systems that you are communicating with.

See Monitoring WebSphere MQ for more information about trace-route messaging in a network.

Is the channel running?

- Issue the following command for the transmission queue:

  ```
  DISPLAY QUEUE (qname) IPPROCS
  ```

  If the value for IPPROCS is 0, this means that the channel serving this transmission queue is not running.
- Issue the following command for the channel:

  ```
  DISPLAY CHSTATUS (channel-name) STATUS MSGS
  ```

  Use the output produced by this command to check that the channel is serving the correct transmission queue and that it is connected to the correct target machine and port. You can determine whether the channel is running from the STATUS field. You can also see if any messages have been sent on the channel by examining the MSGS field.

  If the channel is in RETRYING state, this is probably caused by a problem at the other end. Check that the channel initiator and listener have been started, and that the channel has not been stopped. If somebody has stopped the channel, you need to start it manually.

Is triggering set on in the sending system?

Check that the channel initiator is running.

Does the transmission queue have triggering set on?

If a channel is stopped under specific circumstances, triggering can be set off for the transmission queue.

Is the message you are waiting for a reply message from a remote system?
Check the definitions of the remote system, as described above, and check that triggering is activated in the remote system. Also check that the LU 6.2 connection between the two systems is not single session (if it is, you cannot receive reply messages).

Check that the queue on the remote queue manager exists, is not full, and accepts the message length. If any of these criteria are not fulfilled, the remote queue manager tries to put the message on the dead-letter queue. If the message length is longer than the maximum length that the channel will allow, the sending queue manager tries to put the message on its dead-letter queue.

Is the queue already full?

This could mean that an application could not put the required message on to the queue. If this is so, check if the message has been put on to the dead-letter queue.

The dead-letter queue message header (dead-letter header structure) contains a reason or feedback code explaining why the message could not be put on to the target queue. See the WebSphere MQ Application Programming Reference manual for information about the dead-letter header structure.

Is there a mismatch between the sending and receiving queue managers?
For example, the message length could be longer than the receiving queue manager can handle. Check the console log for error messages.

Are the channel definitions of the sending and receiving channels compatible?
For example, a mismatch in the wrap value of the sequence number will stop the channel. See the WebSphere MQ Intercommunication manual for more information about distributed queuing.

Has data conversion been performed correctly?
If a message has come from a different queue manager, are the CCSIDs and encoding the same, or does data conversion need to be performed.

Has your channel been defined for fast delivery of nonpersistent messages?
If your channel has been defined with the NPMSPEED attribute set to FAST (the default), and the channel has stopped for some reason and then been restarted, nonpersistent messages might have been lost. See the WebSphere MQ Intercommunication manual for more information about fast messages.

Is a channel exit causing the messages to be processed in an unexpected way?
For example, a security exit might prevent a channel from starting, or an *ExitResponse* of MQXCC_CLOSE_CHANNEL might terminate a channel.

Problems with getting messages when using message grouping

Is the application waiting for a complete group of messages?

Ensure all the messages in the group are on the queue. If you are using distributed queuing, see "Problems with missing messages when using distributed queuing" on page 35. Ensure the last message in the group has the appropriate MsgFlags set in the message descriptor to indicate that it is the last message. Ensure the message expiry of the messages in the group is set to a long enough interval that they do not expire before they are retrieved.

If messages from the group have already been retrieved, and the get request is not in logical order, turn off the option to wait for a complete group when retrieving the other group messages.

If the application issues a get request in logical order for a complete group, and midway through retrieving the group it cannot find a message:

Ensure that no other applications are running against the queue and getting messages. Ensure that the message expiry of the messages in the group is set to a long enough interval that they do not expire before they are retrieved. Ensure that no one has issued the CLEAR QUEUE command. You can retrieve incomplete groups from a queue by getting the messages by groupid, without specifying the logical order option.

Finding messages sent to a cluster queue

Before you can use the techniques described in this chapter to find a message that did not arrive at a cluster queue, you need to determine the queue managers that host the queue to which the message was sent. You can determine this in the following ways:

- You can use the DISPLAY QUEUE command to request information about cluster queues.
- You can use the name of the queue and queue manager that is returned in the MQPMO structure.

  If you specified the MQOO_BIND_ON_OPEN option for the message, these fields give the destination of the message. If the message was not bound to a particular queue and queue manager, these fields give the name of the first queue and queue manager to which the message was sent. In this case, this might not be the ultimate destination of the message.

Finding messages sent to the WebSphere MQ-IMS bridge

If you are using the WebSphere MQ-IMS bridge, and your message has not arrived as expected, consider the following:

Is the WebSphere MQ-IMS bridge running?

Issue the following command for the bridge queue:

```
DISPLAY QSTATUS (qname) IPPROCS CURDEPTH
```

The value of IPPROCS should be 1; if it is 0, check the following:

- Is the queue a bridge queue?
- Is IMS running?
- Has OTMA been started?
- Is WebSphere MQ connected to OTMA?

Note: There are two WebSphere MQ messages that you can use to establish whether you have a connection to OTMA. If message CSQ2010I is present in the joblog of the task, but message CSQ2011I is not present, WebSphere MQ is connected to OTMA. This message also tells you to which WebSphere MQ system OTMA is connected. For more information about the content of these messages, refer to the WebSphere MQ Application Programming Reference manual.

Within the queue manager there is a task processing each IMS bridge queue. This task gets from the queue, sends the request to IMS, and then does a commit. If persistent messages are used, then the commit will require disk I/O and so the process will take longer than for non persistent

messages. The time to process the get, send, and commit, limits the rate at which the task can process messages. If the task can keep up with the workload then the current depth will be close to zero. If you find that the current depth is often greater than zero you might be able to increase throughput by using two queues instead of one.

Use the IMS command `/DIS OTMA` to check that OTMA is active.

If your messages are flowing to IMS, check the following:

- Use the IMS command `/DIS TMEMBER client TPIPE ALL` to display information about IMS Tpipes. From this you can determine the number of messages enqueued on, and dequeued from, each Tpipe. (Commit mode 1 messages are not usually queued on a Tpipe.)
- Use the IMS command `/DIS A` to show whether there is a dependent region available for the IMS transaction to run in.
- Use the IMS command `/DIS TRAN trancode` to show the number of messages queued for a transaction.
- Use the IMS command `/DIS PROG progname` to show if a program has been stopped.

Was the reply message sent to the correct place?

Issue the following command:

```
DISPLAY QSTATUS (*) CURDEPTH
```

Does the CURDEPTH indicate that there is a reply on a queue that you are not expecting?

Messages contain unexpected or corrupted information

If the information contained in the message is not what your application was expecting, or has been corrupted in some way, consider the following points:

Has your application, or the application that put the message on to the queue changed?

Ensure that all changes are simultaneously reflected on all systems that need to be aware of the change.

For example, a copybook formatting the message might have been changed, in which case, both applications will have to be recompiled to pick up the changes. If one application has not been recompiled, the data will appear corrupt to the other.

Check that no external source of data, such as a VSAM data set, has changed. This could also invalidate your data if any necessary recompilations have not been done. Also check that any CICS maps and TSO panels that you are using for input of message data have not changed.

Is an application sending messages to the wrong queue?

Check that the messages your application is receiving are not really intended for an application servicing a different queue. If necessary, change your security definitions to prevent unauthorized applications from putting messages on to the wrong queues.

If your application has used an alias queue, check that the alias points to the correct queue.

If you altered the queue to make it a cluster queue, it might now contain messages from different application sources.

Has the trigger information been specified correctly for this queue?

Check that your application should have been started, or should a different application have been started?

Has data conversion been performed correctly?

If a message has come from a different queue manager, are the CCSIDs and encoding the same, or does data conversion need to be performed.

Check that the *Format* field of the MQMD structure corresponds with the content of the message. If not, the data conversion process might not have been able to deal with the message correctly.

If these checks do not enable you to solve the problem, check your application logic, both for the program sending the message, and for the program receiving it.

Chapter 3. Diagnostic aids and techniques

Diagnostic aids

This chapter discusses the following subjects:

- The recovery actions attempted by the queue manager when a problem is detected.
- WebSphere MQ for z/OS abends, and the information produced when an abend occurs.
- The diagnostic information produced by WebSphere MQ for z/OS, and additional sources of useful information.

The type of information provided to help with problem determination and application debugging depends on the type of error encountered, and the way your subsystem is set up.

WebSphere MQ for z/OS recovery actions

WebSphere MQ for z/OS can recover from program checks caused by incorrect user data. A completion and reason code is issued to the caller. These codes are documented in the WebSphere MQ Application Programming Reference manual.

Program errors

Program errors might be associated with user application program code or WebSphere MQ code, and fall into two categories:

- User-detected errors
- Subsystem-detected errors

User-detected errors:

User-detected errors are detected by the user (or a user-written application program) when the results of a service request are not as expected (for example, a nonzero completion code). The collection of problem determination data cannot be automated because detection occurs after the WebSphere MQ function has completed. Rerunning the application with the WebSphere MQ user parameter trace facility activated can provide the data needed to analyze the problem. The output from this trace is directed to the *generalized trace facility* (GTF).

You can turn the trace on and off using an operator command. Refer to "Using trace for problem determination" on page 65 for more information.

Queue manager detected errors:

The queue manager detects errors such as:

- A program check
- A data set filling up
- An internal consistency error

WebSphere MQ analyzes the error and takes the following actions:

- If the problem was caused by a user or application error (such as an invalid address being used), the error is reflected back to the application by completion and reason codes.
- If the problem was not caused by a user or application error (for example, all available DASD has been used, or the system detected an internal inconsistency), WebSphere MQ recovers if possible, either by sending completion and reason codes to the application, or if this is not possible, by abending the application.
- If WebSphere MQ cannot recover, it terminates with a specific reason code. An SVC dump is usually taken recording information in the *system diagnostic work area* (SDWA) and *variable recording area* (VRA) portions of the dump, and an entry is made in SYS1.LOGREC.

WebSphere MQ for z/OS abends

WebSphere MQ for z/OS uses two system abend completion codes, X'5C6' and X'6C6'. These codes identify:

- Internal errors encountered during operation
- Diagnostic information for problem determination
- Actions initiated by the component involved in the error

X'5C6'
An X'5C6' abend completion code indicates that WebSphere MQ has detected an internal error and has terminated an internal task (TCB) or a user-connected task abnormally. Errors associated with a X'5C6' abend completion code might be preceded by a z/OS system code, or by internal errors.

Examine the diagnostic material generated by the X'5C6' abend to determine the source of the error that actually resulted in a subsequent task or subsystem termination.

X'6C6'
An X'6C6' abend completion code indicates that WebSphere MQ has detected a severe error and has terminated the queue manager abnormally. When a X'6C6' is issued, WebSphere MQ has determined that continued operation could result in the loss of data integrity. Errors associated with a X'6C6' abend completion code might be preceded by a z/OS system error, one or more X'5C6' abend completion codes, or by error message CSQV086E indicating abnormal termination of WebSphere MQ.

Table 2 summarizes the actions and diagnostic information available to WebSphere MQ for z/OS when these abend completion codes are issued. Different pieces of this information are relevant in different error situations. The information produced for a given error depends upon the specific problem. The z/OS services that provide diagnostic information are discussed in "WebSphere MQ for z/OS diagnostic information" on page 43.

Table 2. Abend completion codes

	X'5C6'	X'6C6'
Explanation	• Error during WebSphere MQ normal operation	• Severe error; continued operation might jeopardize data integrity

Table 2. Abend completion codes (continued)

	X'5C6'	X'6C6'
System action	• Internal WebSphere MQ task is abended • Connected user task is abended	• The entire WebSphere MQ subsystem is abended • User task with an active WebSphere MQ connection might be abnormally terminated with a X'6C6' code • Possible MEMTERM (memory termination) of connected allied address space
Diagnostic information	• SVC dump • SYS1.LOGREC entry • VRA data entries	• SYS1.LOGREC • VRA data entries
Associated reason codes	• WebSphere MQ abend reason code • Associated z/OS system codes	• Subsystem termination reason code • z/OS system completion codes and X'5C6' codes that precede the X'6C6' abend
Location of accompanying codes	• SVC dump title • Message CSQW050I • Register 15 of SDWA section 'General Purpose Registers at Time of Error' • SYS1.LOGREC entries • VRA data entries	• SYS1.LOGREC • VRA data entries • Message CSQV086E, which is sent to z/OS system operator

WebSphere MQ for z/OS diagnostic information

WebSphere MQ for z/OS functional recovery routines use z/OS services to provide diagnostic information to help you in problem determination.

The following z/OS services provide diagnostic information:

SVC dumps
The WebSphere MQ abend completion code X'5C6' uses the z/OS SDUMP service to create SVC dumps. The content and storage areas associated with these dumps vary, depending on the specific error and the state of the queue manager at the time the error occurred.

SYS1.LOGREC

Entries are requested in the SYS1.LOGREC data set at the time of the error using the z/OS SETRP service. The following are also recorded in SYS1.LOGREC:

- Subsystem abnormal terminations
- Secondary abends occurring in a recovery routine
- Requests from the recovery termination manager

Variable recording area (VRA) data
Data entries are added to the VRA of the SDWA by using a z/OS VRA defined key. VRA data includes a series of diagnostic data entries common to all WebSphere MQ for z/OS abend completion codes. Additional

information is provided during initial error processing by the invoking component recovery routine, or by the recovery termination manager.

z/OS abends

During WebSphere MQ operation, an abend might occur with a z/OS system completion code. If you receive a z/OS abend, see the appropriate z/OS publication.

CICS abends

A CICS abend message is sent to the terminal, if the application is attached to one, or to the CSMT log. CICS abend codes are explained in the *CICS Messages and Codes* manual.

The CICS adapter issues abend reason codes beginning with the letter 'Q' (for example, QDCL). These codes are documented in the WebSphere MQ for z/OS Messages and Codes manual.

IMS abends

An IMS application might abend in one of the following circumstances:

- A normal abend.
- An IMS pseudo abend, with an abend code such as U3044 resulting from an error in an ESAF exit program.
- Abend 3051 or 3047, when the REO (region error option) has been specified as 'Q' or 'A', and an IMS application attempts to reference a non-operational external subsystem, or when resources are unavailable at the time when a thread is created.

An IMS message is sent to the user terminal or job output, and the IMS master terminal. The abend might be accompanied by a region dump.

Diagnostic information produced

WebSphere MQ for z/OS provides unique messages that, together with the output of dumps, are aimed at providing sufficient data to allow diagnosis of the problem without having to try to reproduce it. This is known as first failure data capture.

Error messages

WebSphere MQ produces an error message when a problem is detected. WebSphere MQ diagnostic messages begin with the prefix CSQ. Each error message generated by WebSphere MQ is unique; that is, it is generated for one and only one error. Information about the error can be found in the WebSphere MQ for z/OS Messages and Codes manual.

The first three characters of the names of WebSphere MQ modules are also usually CSQ. The exceptions to this are modules for C++ (IMQ), and the header files (CMQ). The fourth character uniquely identifies the component. These identifiers are listed in Chapter 7, "WebSphere MQ component and resource manager identifiers," on page 101. Characters five through eight are unique within the group identified by the first four characters.

Make sure that you have some documentation on application messages and codes for programs that were written at your installation, as well as a copy of the WebSphere MQ for z/OS Messages and Codes manual.

There might be some instances when no message is produced, or, if one is produced, it cannot be communicated. In these circumstances, you might have to analyze a dump to isolate the error to a particular module. The use of dumps is discussed in "WebSphere MQ dumps" on page 48.

Dumps

Dumps are an important source of detailed information about problems. Whether they are as the result of an abend or a user request, they allow you to see a "snapshot" of what was happening at the moment the dump was taken. "WebSphere MQ dumps" on page 48 contains guidance about using dumps to locate problems in your WebSphere MQ system. However, because they only provide a "snapshot", you might need to use them in conjunction with other sources of information that cover a longer period of time, such as logs.

Snap dumps are also produced for specific types of error in handling MQI calls. The dumps are written to the CSQSNAP DD.

Console logs and job output

You can copy console logs into a permanent data set, or print them as required. If you are only interested in specific events, you can select which parts of the console log to print.

Job output includes output produced from running the job, as well as that from the console. You can copy this into permanent data sets, or print it as required. You might need to collect output for all associated jobs, for example CICS, IMS, and WebSphere MQ.

Symptom strings

Symptom strings display important diagnostic information in a structured format. When a symptom string is produced, it is available in one or more of the following places:

- On the z/OS system console
- In SYS1.LOGREC
- In any dump taken

Figure 1 shows an example of a symptom string.

```
PIDS/5655L8200 RIDS/CSQMAIN1 AB/S6C6 PRCS/0E30003
```

Figure 1. Sample symptom string

The symptom string provides a number of keywords that you can use to search the IBM software support database. If you have access to one of the optional search tools, you can search the database yourself. If you report a problem to the IBM support center, you are often asked to quote the symptom string. (For more information about searching the IBM software support database, refer to Chapter 4, "Finding solutions to similar problems," on page 75.)

Although the symptom string is really designed to provide keywords for searching the database, it can also give you a lot of information about what was happening at the time the error occurred, and it might suggest an obvious cause or a promising area to start your investigation. See "Building a keyword string" on page 78 for more information about keywords.

Queue information

You can display information about the status of queues by using the operations and control panels. Alternatively you can enter the DISPLAY QUEUE and DISPLAY QSTATUS commands from the z/OS console.

Note: If the command was issued from the console, the response is copied to the console log, allowing the documentation to be kept together compactly.

Other sources of information

You might find the following items of documentation useful when solving problems with WebSphere MQ for z/OS.

Your own documentation

Your own documentation is the collection of information produced by your organization about what your system and applications should do, and how they are supposed to do it. How much of this kind of information you need depends on how familiar you are with the system or application in question, and could include:

- Program descriptions or functional specifications
- Flowcharts or other descriptions of the flow of activity in a system
- Change history of a program
- Change history of your installation
- Statistical and monitoring profile showing average inputs, outputs, and response times

Manuals for the products you are using

The manuals for the product you are using are the books in the WebSphere MQ library, and in the libraries for any other products you use with your application.

Make sure that the level of any book you refer to matches the level of the system you are using. Problems often arise through using either obsolete information, or information about a level of a product that is not yet installed.

Source listings and link-edit maps

Include the source listings of any applications written at your installation with your set of documentation. (They can often be the largest single element of documentation. Large installations with thousands of programs often keep such listings on microfiche.) Make sure you include the relevant output from the linkage editor with your source listings to avoid wasting time trying to find your way through a load module with an out-of-date link map. Be sure to include the JCL at the beginning of your listings, to show the libraries that were used and the load library the load module was placed in.

Change log

The information in the change log can tell you of changes made in the data processing environment that might have caused problems with your application program. To get the most out of your change log, include the data concerning hardware changes, system software (such as z/OS and WebSphere MQ) changes, application changes, and any modifications made to operating procedures.

System configuration charts

System configuration charts show what systems are running, where they are running, and how the systems are connected to each other. They also show which WebSphere MQ, CICS, or IMS systems are test systems and which are production systems.

Information from the DISPLAY CONN command

The DISPLAY CONN command provides information about which applications are connected to a queue manager, and information to help you to diagnose those that have a long-running unit of work. You could collect this information periodically and check it for any long-running units of work, and display the detailed information about that connection.

Diagnostic aids for CICS

You can use the CKQC transaction (the CICS adapter control panels) to display information about queue manager tasks, and what state they are in (for example, a GET WAIT). See the WebSphere MQ for z/OS System Administration Guide for information about CKQC.

The application development environment is the same as for any other CICS application, and so you can use any tools normally used in that environment to develop WebSphere MQ applications. In particular, the *CICS execution diagnostic facility* (CEDF) traps entry to and exit from the CICS adapter for each MQI call, as well as trapping calls to all CICS API services. Examples of the output produced by this facility are given in Chapter 9, "Examples of CEDF output," on page 107.

The CICS adapter also writes trace entries to the CICS trace. These entries are described in Chapter 8, "CICS adapter trace entries," on page 103.

Additional trace and dump data is available from the CICS region. These entries are as described in the *CICS Problem Determination Guide*.

Diagnostic aids for IMS

The application development environment is the same as for any other IMS application, and so any tools normally used in that environment can be used to develop WebSphere MQ applications.

Trace and dump data is available from the IMS region. These entries are as described in the *IMS/ESA Diagnosis Guide and Reference* manual.

Diagnostic aids for DB2

Refer to the following manuals for help in diagnosing DB2 problems:

- *DB2 for z/OS Diagnosis Guide and Reference*
- *DB2 Messages and Codes*

WebSphere MQ dumps

This chapter discusses the use of dumps in problem determination. It describes the steps you should take when looking at a dump produced by a WebSphere MQ for z/OS address space. The following topics are discussed:

- "How to use dumps for problem determination"
- "Getting a dump"
- "Processing a dump" on page 51
- "Analyzing the dump" on page 61
- "SYSUDUMP information" on page 62
- "SYS1.LOGREC information" on page 64
- "When SVC dumps are not produced" on page 64

How to use dumps for problem determination

When solving problems with your WebSphere MQ for z/OS system, you can use dumps in two ways:

- To examine the way WebSphere MQ processes a request from an application program.

  To do this, you usually need to analyze the whole dump, including control blocks and the internal trace.
- To identify problems with WebSphere MQ for z/OS itself, under the direction of IBM support center personnel.

You will often find that the dump title provides sufficient information in the abend and reason codes to resolve the problem. You can see the dump title in the console log, or by using the z/OS command `DISPLAY DUMP,TITLE`. The format of the dump title is explained in "Analyzing the dump" on page 61. WebSphere MQ for z/OS abend codes are discussed in "WebSphere MQ for z/OS abends" on page 42, and abend reason codes are documented in the WebSphere MQ for z/OS Messages and Codes manual.

If there is not enough information about your problem in the dump title, format the dump to display the other information contained in it.

Getting a dump

The following table shows information about the types of dump used with WebSphere MQ for z/OS and how they are initiated. It also shows how the dump is formatted:

Table 3. Types of dump used with WebSphere MQ for z/OS

Dump type	Data set	Output type	Formatted by	Caused by
SVC	Defined by system	Machine readable	IPCS in conjunction with a WebSphere MQ for z/OS verb exit	z/OS or WebSphere MQ for z/OS functional recovery routine detecting error, or the operator entering the z/OS DUMP command
SYSUDUMP	Defined by JCL (SYSOUT=A)	Formatted	Normally SYSOUT=A	An abend condition (only taken if there is a SYSUDUMP DD statement for the step)
Snap	Defined by JCL CSQSNAP (SYSOUT=A)	Formatted	Normally SYSOUT=A	Unexpected MQI call errors reported to adapters, or FFST™ information from the channel initiator
Stand-alone	Defined by installation (tape or disk)	Machine readable	IPCS in conjunction with a WebSphere MQ for z/OS verb exit	Operator IPL of the stand-alone dump program

WebSphere MQ for z/OS recovery routines request SVC dumps for most X'5C6' abends. The exceptions are listed in "When SVC dumps are not produced" on page 64. SVC dumps issued by WebSphere MQ for z/OS are the primary source of diagnostic information for problems.

If the dump is initiated by the WebSphere MQ subsystem, information about the dump is put into area called the *summary portion*. This contains information that the dump formatting program can use to identify the key components.

For more information about SVC dumps, see the *MVS Diagnosis: Tools and Service Aids* manual.

Using the z/OS DUMP command

You might be asked to dump any or several of the following for IBM to resolve the problem:

- Main WebSphere MQ address space
- Channel initiator address space
- Coupling Facility application structure
- Coupling Facility administration structure for your queue-sharing group

Figure 2 on page 50 through to Figure 6 on page 51 show examples of the z/OS commands to do this, assuming a subsystem name of CSQ1.

```
DUMP COMM=(MQ QUEUE MANAGER DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=(CSQ1MSTR,BATCH,MASTER),CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
 IEE600I REPLY TO 01 IS;JOBNAME=CSQ1MSTR,CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
 IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=MQ QUEUE MANAGER MAIN DUMP
```

Figure 2. Dumping the WebSphere MQ queue manager and application address spaces

```
DUMP COMM=(MQ QUEUE MANAGER DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=(CSQ1MSTR,MASTER),CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
 IEE600I REPLY TO 01 IS;JOBNAME=CSQ1MSTR,CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
 IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=MQ QUEUE MANAGER DUMP
```

Figure 3. Dumping the WebSphere MQ queue manager address space

```
DUMP COMM=(MQ CHIN DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=CSQ1CHIN,CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
 IEE600I REPLY TO 01 IS;JOBNAME=CSQ1CHIN,CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),CONT
*03 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
 IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),CONT
R 03,DSPNAME=('CSQ1CHIN'.CSQXTRDS),END
IEE600I REPLY TO 03 IS;DSPNAME='CSQ1CHIN'.CSQXTRDS,END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=MQ CHIN DUMP
```

Figure 4. Dumping the channel initiator address space

```
DUMP COMM=(MQ MSTR & CHIN DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=(CSQ1MSTR,CSQ1CHIN),CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
 IEE600I REPLY TO 01 IS;JOBNAME=(CSQ1MSTR,CSQ1CHIN),CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),CONT
*03 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
 IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,LSQA,TRT,SUM),CONT
R 03,DSPNAME=('CSQ1CHIN'.CSQXTRDS),END
IEE600I REPLY TO 03 IS;DSPNAME=('CSQ1CHIN'.CSQXTRDS),END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=MQ MSTR & CHIN DUMP
```

Figure 5. Dumping the WebSphere MQ queue manager and channel initiator address spaces

```
DUMP COMM=('MQ APPLICATION STRUCTURE 1 DUMP')
01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,STRLIST=(STRNAME=QSG1APPLICATION1,(LISTNUM=ALL,ADJUNCT=CAPTURE,ENTRYDATA=UNSER))
IEE600I REPLY TO 01 IS;STRLIST=(STRNAME=QSG1APPLICATION1,(LISTNUM=
IEA794I SVC DUMP HAS CAPTURED: 677
DUMPID=057 REQUESTED BY JOB (*MASTER*)
DUMP TITLE='MQ APPLICATION STRUCTURE 1 DUMP'
```

Figure 6. Dumping a Coupling Facility structure

Note: The examples shown in Figure 3 on page 50, Figure 4 on page 50, and Figure 5 on page 50 relate to z/OS only. For MVS, specify ALLPSA instead of PSA and SUMDUMP instead of SUM.

Processing a dump

You can process a dump in several ways:

- Use the IPCS panels provided by WebSphere MQ for z/OS, as described in "Using the WebSphere MQ for z/OS dump display panels."
- Use the IPCS dialog under TSO using line mode IPCS commands, as described in "Using line mode IPCS" on page 55
- Use IPCS as a batch job to produce the output as a print file, as described in "Using IPCS in batch" on page 60

Using the WebSphere MQ for z/OS dump display panels

WebSphere MQ for z/OS provides a set of panels to help you process dumps. The following section describes how to use these panels:

1. From the IPCS PRIMARY OPTION MENU, select **ANALYSIS - Analyze dump contents** (option 2).

   The IPCS MVS ANALYSIS OF DUMP CONTENTS panel appears.
2. Select **COMPONENT - MVS component data** (option 6).

   The IPCS MVS DUMP COMPONENT DATA ANALYSIS panel appears. The appearance of the panel depends on the products installed at your installation, but will be similar to the panel shown below:

```
---------------- IPCS MVS DUMP COMPONENT DATA ANALYSIS -------------
OPTION ===>                                              SCROLL ===

To display information, specify "S option name" or enter S to the
left of the option desired.  Enter ? to the left of an option to
display help regarding the component support.

  Name     Abstract
  ALCWAIT  Allocation wait summary
  AOMDATA  AOM analysis
  ASMCHECK Auxiliary storage paging activity
  ASMDATA  ASM control block analysis
  AVMDATA  AVM control block analysis
  COMCHECK Operator communications data
  CSQMAIN  WebSphere MQ dump formatter panel interface
  CSQWDMP  WebSphere MQ dump formatter
  CTRACE   Component trace summary
  DAEDATA  DAE header data
  DIVDATA  Data-in-virtual storage
```

3. Select **CSQMAIN WebSphere MQ dump formatter panel interface** by typing s beside the line and pressing Enter.

If this option is not available, it is because the member CSQ7IPCS is not present; you should see the WebSphere MQ for z/OS System Setup Guide for information about installing the WebSphere MQ for z/OS dump formatting member.

Note: If you have already used the dump to do a preliminary analysis, and you want to reexamine it, select **CSQWDMP WebSphere MQ dump formatter** to display the formatted contents again, using the default options.

4. The IBM WebSphere MQ for z/OS - DUMP ANALYSIS menu appears. Use this menu to specify the action that you want to perform on a system dump.

```
---------------IBM WebSphere MQ for z/OS - DUMP ANALYSIS----------------
 COMMAND ===>



      1 Display all dump titles 00 through 99
      2 Manage the dump inventory
      3 Select a dump

      4 Display address spaces active at time of dump
      5 Display the symptom string
      6 Display the symptom string and other related data
      7 Display LOGREC data from the buffer in the dump
      8 Format and display the dump

      9 Issue IPCS command or CLIST



(c) Copyright IBM Corporation 1993, 2005. All rights reserved.

  F1=Help    F3=Exit   F12=Cancel
```

5. Before you can select a particular dump for analysis, the dump you require must be present in the dump inventory. To ensure that this is so, perform the following steps:
   a. If you do not know the name of the data set containing the dump, specify option 1 - **Display all dump titles xx through xx**.

      This displays the dump titles of all the dumps contained in the SYS1.DUMP data sets (where xx is a number in the range 00 through 99). You can limit the selection of data sets for display by using the xx fields to specify a range of data set numbers.

      If you want to see details of all available dump data sets, set these values to 00 and 99.

      Use the information displayed to identify the dump you want to analyze.
   b. If the dump has not been copied into another data set (that is, it is in one of the SYS1.DUMP data sets), specify option 2 - **Manage the dump inventory**

      The dump inventory contains the dump data sets that you have used. Because the SYS1.DUMP data sets are reused, the name of the dump that you identified in step 5a might appear in the list displayed. However, this entry refers to the previous dump that was stored in this data set, so delete it by typing DD next to it and pressing Enter. Then press F3 to return to the DUMP ANALYSIS MENU.
6. Specify option 3 - **Select a dump**, to select the dump that you want to work with. Type the name of the data set containing the dump in the Source field, check that NOPRINT and TERMINAL are specified in the Message Routing field (this is to ensure that the output is directed to the terminal), and press Enter. Press F3 to return to the DUMP ANALYSIS MENU.

7. Having selected a dump to work with, you can now use the other options on the menu to analyze the data in different parts of the dump:
   - To display a list of all address spaces active at the time the dump was taken, select option 4.
   - To display the symptom string, select option 5. If you want to use the symptom string to search the RETAIN® database for solutions to similar problems, refer to "Searching the IBM database" on page 75.
   - To display the symptom string and other serviceability information, including the variable recording area of the system diagnostic work area (SDWA), select option 6.
   - To format and display the data contained in the in-storage LOGREC buffer, select option 7.

     It could be that the abend that caused the dump was not the original cause of the error, but was caused by an earlier problem. To determine which LOGREC record relates to the cause of the problem, go to the bottom of the data set, type `FIND ERRORID: PREV`, and press Enter. The header of the latest LOGREC record is displayed, for example:

```
JOBNAME: NONE-FRR
ERRORID: SEQ=00081  CPU=0040  ASID=0033  TIME=14:42:47.1

SEARCH ARGUMENT ABSTRACT

  PIDS/5655L8200 RIDS/CSQRLLM1#L RIDS/CSQRRHSL AB/S05C6
  PRCS/00D10231 REGS/0C1F0 RIDS/CSQVEUS2#R

  SYMPTOM             DESCRIPTION
  -------             -----------
  PIDS/5655L8200      PROGRAM ID: 5655L8200
  .
  .
  .
```

     Note the program identifier (if it is not 5655L8200, the problem was not caused by WebSphere MQ for z/OS and you could be looking at the wrong dump). Also note the value of the TIME field. Repeat the command to find the previous LOGREC record, and note the value of the TIME field again. If the two values are close to each other (say, within about one or two tenths of a second), they could both relate to the same problem.

     You can use the symptom string from the LOGREC record related to the error to search the RETAIN database for solutions to similar problems (refer to "Searching the IBM database" on page 75).
   - To format and display the dump, select option 8. The FORMAT AND DISPLAY THE DUMP panel appears:

```
---------IBM WebSphere MQ for z/OS - FORMAT AND DISPLAY DUMP--------
 COMMAND ===>

    1  Display the control blocks and trace
    2  Display just the control blocks
    3  Display just the trace


 Options:

 Use the summary dump? . . . . . . . . . . . . . . . __ 1 Yes
                                                       2 No


 Subsystem name (required if summary dump not used) ____


 Address space identifier or ALL. . . . . . . . . . ALL_



  F1=Help    F3=Exit   F12=Cancel
```

- Use this panel to format your selected system dump. You can choose to display control blocks, data produced by the internal trace, or both, which is the default.

  Note: You cannot do this for dumps from the channel initiator, or for dumps of Coupling Facility structures.

  – To display the whole of the dump, that is:
    - The dump title
    - The variable recording area (VRA) diagnostic information report
    - The save area trace report
    - The control block summary
    - The trace table

    select option 1.

  – To display the information listed for option 1, without the trace table, select option 2.

  – To display the information listed for option 1, without the control blocks, select option 3.

  You can also use the following options:

  – **Use the Summary Dump?**

    Use this field to specify whether you want WebSphere MQ to use the information contained in the summary portion when formatting the selected dump. The default setting is YES.

    Note: If a summary dump has been taken, it might include data from more than one address space.

  – **Subsystem name**

    Use this field to identify the subsystem whose dump data you want to display. This is only required if there is no summary data (for example, if the operator requested the dump), or if you have specified NO in the **Use the summary dump?** field.

    If you do not know the subsystem name, type `IPCS SELECT ALL` at the command prompt, and press Enter to display a list of all the jobs running at the time of the error. If one of the jobs has the word ERROR against it

in the SELECTION CRITERIA column, make a note of the name of that job. The job name is of the form *xxxx*MSTR, where *xxxx* is the subsystem name.

```
IPCS OUTPUT STREAM ------------------------
 COMMAND ===>
   ASID JOBNAME  ASCBADDR  SELECTION CRITERIA
   ---- -------- --------  ------------------
   0001 *MASTER* 00FD4D80  ALL
   0002 PCAUTH   00F8AB80  ALL
   0003 RASP     00F8C100  ALL
   0004 TRACE    00F8BE00  ALL
   0005 GRS      00F8BC00  ALL
   0006 DUMPSRV  00F8DE00  ALL
   0008 CONSOLE  00FA7E00  ALL
   0009 ALLOCAS  00F8D780  ALL
   000A SMF      00FA4A00  ALL
   000B VLF      00FA4800  ALL
   000C LLA      00FA4600  ALL
   000D JESM     00F71E00  ALL
   001F MQM1MSTR 00FA0680  ERROR ALL
```

If no job has the word ERROR against it in the SELECTION CRITERIA column, select option 0 - DEFAULTS on the main IPCS Options Menu panel to display the IPCS Default Values panel. Note the address space identifier (ASID) and press F3 to return to the previous panel. Use the ASID to determine the job name; the form is *xxxx*MSTR, where *xxxx* is the subsystem name.

The following command shows which ASIDs are in the dump data set:

```
LDMP DSN('SYS1.DUMPxx') SELECT(DUMPED) NOSUMMARY
```

This shows the storage ranges dumped for each address space.

Press F3 to return to the FORMAT AND DISPLAY THE DUMP panel, and type this name in the **Subsystem name** field.

– **Address space identifier**

Use this field if the data in a dump comes from more than one address space. If you only want to look at data from a particular address space, specify the identifier (ASID) for that address space.

The default value for this field is ALL, which displays information about all the address spaces relevant to the subsystem in the dump. Change this field by typing the 4-character ASID over the value displayed.

Note: Because the dump contains storage areas common to all address spaces, the information displayed might not be relevant to your problem if you specify the address space identifier incorrectly. In this case, return to this panel, and enter the correct address space identifier.

Using line mode IPCS

To format the dump using line mode IPCS commands, select the dump required by issuing the command:

```
SETDEF DSN('SYS1.DUMPxx')
```

(where `SYS1.DUMPxx` is the name of the data set containing the dump). You can then use IPCS subcommands to display data from the dump.

Formatting a WebSphere MQ for z/OS dump:

The IPCS VERBEXIT CSQWDMP invokes the WebSphere MQ for z/OS dump formatting program (CSQWDPRD), and enables you to format an SVC dump to display WebSphere MQ data. You can restrict the amount of data that is displayed by specifying parameters.

See page "Formatting a Coupling Facility structure dump" on page 60 for details on formatting a Coupling Facility list structure.

Note: This section describes the parameters required to extract the necessary data. Separate operands by commas, not blanks. A blank that follows any operand in the control statement terminates the operand list, and any subsequent operands are ignored. Table 4 explains each keyword that you can specify in the control statement for formatting dumps.

Table 4. Keywords for the WebSphere MQ for z/OS dump formatting control statement

Keyword	Description
SUBSYS=*aaaa*	Use this keyword if the summary dump portion is not available, or not to be used, to give the name of the subsystem to format information for. *aaaa* is a 1 through 4-character subsystem name.
ALL (default)	All control blocks and the trace table.
AA	Data is displayed for all WebSphere MQ for z/OS control blocks in all address spaces.
DIAG=Y	Print diagnostic information. Use only under guidance from IBM service personnel. DIAG=N (suppresses the formatting of diagnostic information) is the default.
EB=*nnnnnnnn*	Only the trace points associated with this EB thread are displayed (the format of this keyword is EB=*nnnnnnnn* where *nnnnnnnn* is the 8–digit address of an EB thread that is contained in the trace). You must use this in conjunction with the TT keyword.
LG	All control blocks.
PTF=Y, LOAD=*load module name*	A list of PTFs at the front of the report (from MEPL). PTF=N (suppresses the formatting of such a list) is the default. The optional load subparameter allows you to specify the name of a load module, up to a maximum of 8 characters, for which to format a PTF report.
SA=*hhhh*	The control blocks for a specified address space. Use either of the following formats: • SA=*hh* or • SA=*hhhh* where *h* represents a hexadecimal digit.
SG	A subset of system-wide control blocks.
TT	The trace table only.

Table 5 on page 57 details the dump formatting keywords that you can use to format the data relating to individual resource managers. You cannot use these keywords in conjunction with any of the keywords in Table 4.

Table 5. Resource manager dump formatting keywords

Keyword	What is formatted
BMC=1 BMC=2(*buffer pool number*)	Buffer manager data. BMC=1 formats control blocks of all buffers. BMC=2 formats data relating to the buffer identified in the 2-digit *buffer pool number*.
CFS=1	Coupling Facility manager report.
DB2=1	DB2 manager report.
DMC=1, ONAM=*object name*	Data manager data. The optional ONAM subparameter allows you to specify the object name, up to a maximum of 20 characters, to limit data printed to objects starting with characters in ONAM.
DMC=2, ONAM=*object name*	Formats scavenger information. The optional ONAM subparameter allows you to limit the objects printed to those with names beginning with the characters specified in ONAM (up to a maximum of 20 characters).
IMS=1	IMS bridge data.
LMC=1	Log manager data.
MMC=1, OBJ=*object type*, ONAM=*object name*	Message manager data. The optional OBJ subparameter allows you to select the message manager data for a specific object type. Permitted *object type* values are: • MAUT (for authentication information) • MQLO (for local queues that are not shared) • MQSH (for local queues that are shared) • MQRO (for remote queues) • MQAO (for alias queues) • MQMO (for model queues) • MCHL (for channels) • MNLS (for namelists) • MSTC (for storage classes) • MPRC (for processes) The optional ONAM subparameter allows you to limit the objects printed to those with names beginning with the characters specified in ONAM (up to a maximum of 20 characters).
MMC=2, ONAM=*object name*	MMC=2 formats trigger information for local queues. The optional ONAM subparameter allows you to limit the objects printed to those with names beginning with the characters specified in ONAM (up to a maximum of 20 characters).
THR=*	A list of all threads with connections.
THR=*EB_address*	Thread-related data for the specified EB thread.

If the dump is initiated by the operator, there is no information in the summary portion of the dump. Table 6 on page 58 shows additional keywords that you can use in the CSQWDMP control statement.

Table 6. Summary dump keywords for the WebSphere MQ for z/OS dump formatting control statement

Keyword	Description
SUBSYS=*aaaa*	Use this keyword if the summary dump portion is not available, or not to be used, to give the name of the subsystem to format information for. *aaaa* is a 1 through 4-character subsystem name.
SUMDUMP=NO	Use this keyword if the dump has a summary portion, but you do not want to use it. (You would usually only do this if so directed by your IBM support center.)

The following list shows some examples of how to use these keywords:

- For default formatting of all address spaces, using information from the summary portion of the dump, use:

  ```
  VERBX CSQWDMP
  ```

- To display the trace table from a dump of subsystem named MQMT, which was initiated by an operator (and so does not have a summary portion) use:

  ```
  VERBX CSQWDMP 'TT,SUBSYS=MQMT'
  ```

- To display all the control blocks and the trace table from a dump produced by a subsystem abend, for an address space with ASID (address space identifier) 1F, use:

  ```
  VERBX CSQWDMP 'TT,LG,SA=1F'
  ```

- To display the portion of the trace table from a dump associated with a particular EB thread, use:

  ```
  VERBX CSQWDMP 'TT,EB=nnnnnnnn'
  ```

- To display message manager 1 report for local non-shared queue objects whose name begins with 'ABC' use:

  ```
  VERBX CSQWDMP 'MMC=1,ONAM=ABC,Obj=MQLO'
  ```

Table 7 shows some other commands that are used frequently for analyzing dumps. For more information about these subcommands, see the *MVS IPCS Commands* manual.

Table 7. IPCS subcommands used for dump analysis

Subcommand	Description
STATUS	To display data usually examined during the initial part of the problem determination process.
STRDATA LISTNUM(ALL) ENTRYPOS(ALL) DETAIL	To format Coupling Facility structure data.
VERBEXIT LOGDATA	To format the in-storage LOGREC buffer records present before the dump was taken. LOGDATA locates the LOGREC entries that are contained in the LOGREC recording buffer and invokes the EREP program to format and print the LOGREC entries. These entries are formatted in the style of the normal detail edit report.
VERBEXIT TRACE	To format the system trace entries for all address spaces.
VERBEXIT SYMPTOM	To format the symptom strings contained in the header record of a system dump such as stand-alone dump, SVC dump, or an abend dump requested with a SYSUDUMP DD statement.
VERBEXIT GRSTRACE	To format diagnostic data from the major control blocks for global resource serialization.

Table 7. IPCS subcommands used for dump analysis (continued)

Subcommand	Description
VERBEXIT SUMDUMP	To locate and display the summary dump data that an SVC dump provides.
VERBEXIT DAEDATA	To format the dump analysis and elimination (DAE) data for the dumped system.

Formatting a dump from the channel initiator:

The IPCS VERBEXIT CSQXDPRD enables you to format a channel initiator dump. You can select the data that is formatted by specifying keywords.

This section describes the keywords that you can specify.

Table 8 describes the keywords that you can specify with CSQXDPRD.

Table 8. Keywords for the IPCS VERBEXIT CSQXDPRD

Keyword	What is formatted
SUBSYS=*aaaa*	The control blocks of the channel initiator associated with the named subsystem. It is required for all new formatted dumps.
CHST=1, CNAM=*channel name*, DUMP=S|F|C	All channel information. The optional CNAM subparameter allows you to specify the name of a channel, up to a maximum of 20 characters, for which to format details. The optional DUMP subparameter allows you to control the extent of formatting, as follows: • Specify DUMP=S (for "short") to format the first line of the hexadecimal dump of the channel data. • Specify DUMP=F (for "full") to format all lines of the data. • Specify DUMP=C (for "compressed") to suppress the formatting of all duplicate lines in the data containing only X'00'. This is the default option
CHST=2, CNAM=*channel name*,	A summary of all channels, or of the channel specified by the CNAM keyword. See CHST=1 for details of the CNAM subparameter.
CHST=3, CNAM=*channel name*,	Data provided by CHST=2 and a program trace, line trace and formatted semaphore table print of all channels in the dump. See CHST=1 for details of the CNAM subparameter.
CLUS=1	Cluster report including the cluster repository known on the queue manager.
CLUS=2	Cluster report showing cluster registrations.

Table 8. Keywords for the IPCS VERBEXIT CSQXDPRD (continued)

Keyword	What is formatted
CTRACE=S|F, DPRO=*nnnnnnnn*, TCB=*nnnnnnn*	Select either a short (CTRACE=S) or full (CTRACE=F) CTRACE. The optional DPRO subparameter allows you to specify a CTRACE for the DPRO specified. The optional TCB subparameter allows you to specify a CTRACE for the job specified.
DISP=1, DUMP=S|F|C	Dispatcher report See CHST=1 for details of the DUMP subparameter.
BUF=1	Buffer report

Formatting a Coupling Facility structure dump:

The Coupling Facility administration structure and application structures for your queue-sharing group, in conjunction with dumps of queue managers in the queue-sharing group might be required by IBM Service Personnel to aid problem diagnosis.

For information on formatting a Coupling Facility list structure, and the STRDATA subcommand, see the *MVS IPCS Commands* book.

Using IPCS in batch

To use IPCS in batch, insert the required IPCS statements into your batch job stream (see Figure 7).

Change the data set name (DSN=) on the DUMP00 statement to reflect the dump you want to process, and insert the IPCS subcommands that you want to use.

```
//*************************************************
//*   RUNNING IPCS IN A BATCH JOB                 *
//*************************************************
//MQMDMP   EXEC  PGM=IKJEFT01,REGION=5120K
//STEPLIB  DD    DSN=mqm.library-name,DISP=SHR
//SYSTSPRT DD    SYSOUT=*
//IPCSPRNT DD    SYSOUT=*
//IPCSDDIR DD    DSN=dump.directory-name,DISP=OLD
//DUMP00   DD    DSN=dump.name,DISP=SHR
//SYSTSIN  DD    *
  IPCS NOPARM TASKLIB(SCSQLOAD)
  SETDEF PRINT TERMINAL DDNAME(DUMP00) NOCONFIRM
  ************************************************
  * INSERT YOUR IPCS COMMANDS HERE, FOR EXAMPLE:   *
  VERBEXIT LOGDATA
  VERBEXIT SYMPTOM
  VERBEXIT CSQWDMP 'TT,SUBSYS=QMGR'
  ************************************************

  CLOSE   ALL
END
/*
```

Figure 7. Sample JCL for printing dumps through IPCS in the z/OS environment

Analyzing the dump

The dump title includes the abend completion and reason codes, the failing load module and CSECT names, and the release identifier.

The formats of SVC dump titles vary slightly, depending on the type of error.

Figure 8 shows an example of an SVC dump title. Each field in the title is described after the figure.

```
ssnm,ABN=5C6-00D303F2,U=AUSER,C=R3600.700.LOCK-CSQL1GET,
 M=CSQGFRCV,LOC=CSQLLPLM.CSQL1GET+0246
```

Figure 8. Sample SVC dump title

ssnm,ABN=compltn-reason

- `ssnm` is the name of the subsystem that issued the dump.
- `compltn` is the 3-character hexadecimal abend completion code (in this example, X'5C6'), prefixed by U for user abend codes.
- `reason` is the 4-byte hexadecimal reason code (in this example, X'00D303F2').

Note: The abend and reason codes might provide sufficient information to resolve the problem. See the WebSphere MQ for z/OS Messages and Codes manual for an explanation of the reason code.

U=userid

- `userid` is the user identifier of the user (in this example, AUSER). This field is not present for channel initiators.

C=compid.release.comp-function

- `compid` is the last 5 characters of the component identifier (explained in "Component-identifier keyword" on page 81). The value R3600 uniquely identifies WebSphere MQ for z/OS.
- `release` is a 3-digit code indicating the version, release, and modification level of WebSphere MQ for z/OS (in this example, 700).
- `comp` is an acronym for the component in control at the time of the abend (in this example, LOCK).
- `function` is the name of a function, macro, or routine in control at the time of abend (in this example, CSQL1GET). This field is not always present.

M=module

- `module` is the name of the FRR or ESTAE recovery routine (in this example, CSQGFRCV). This field is not always present.

  Note: This is not the name of the module where the abend occurred; that is given by `LOC`.

LOC=loadmod.csect+csect_offset

- `loadmod` is the name of the load module in control at the time of the abend (in this example, CSQLLPLM). This might be represented by an asterisk if it is unknown.
- `csect` is the name of the CSECT in control at the time of abend (in this example, CSQL1GET).

- `csect_offset` is the offset within the failing CSECT at the time of abend (in this example, 0246).

Note: The value of `csect_offset` might vary if service has been applied to this CSECT, so do not use this value when building a keyword string to search the IBM software support database.

Dump title variation with PSW and ASID

Some dump titles replace the load module name, CSECT name, and CSECT offset with the PSW (program status word) and ASID (address space identifier). Figure 9 illustrates this format.

```
ssnm,ABN=compltn-reason,U=userid,C=compid.release.comp-function,
 M=module,PSW=psw_contents,ASID=address_space_id
```

Figure 9. Dump title with PSW and ASID

psw_contents

- The PSW at the time of the error (for example, X'077C100000729F9C').

address_space_id

- The address space in control at the time of the abend (for example, X'0011'). This field is not present for a channel initiator.

SYSUDUMP information

SYSUDUMP dumps provide information useful for debugging batch and TSO application programs. For more information about SYSUDUMP dumps, see the *MVS Diagnosis: Tools and Service Aids* manual.

Figure 10 on page 63 shows a sample of the beginning of a SYSUDUMP dump.

```
JOB MQMBXBA1  STEP TSOUSER  TIME 102912   DATE 001019   ID = 000  CPUID = 632202333081   PAGE 00000001

COMPLETION CODE      SYSTEM = 0C1      REASON CODE = 00000001

PSW AT ENTRY TO ABEND  078D1000 000433FC        ILC 2  INTC 000D

PSW LOAD MODULE = BXBAAB01  ADDRESS = 000433FC  OFFSET = 0000A7F4

ASCB: 00F56400
+0000 ASCB..... ASCB      FWDP..... 00F60180  BWDP..... 0047800  CMSF..... 019D5A30  SVRB..... 008FE9E0
+0014 SYNC..... 00000D6F  IOSP..... 00000000  TNEW..... 00D18F0  CPUS..... 00000001  ASID..... 0066
+0026 R026..... 0000      LL5...... 00        HLHI..... 01       DPHI..... 00        DP....... 9D
+002C TRQP..... 80F5D381  LDA...... 7FF154E8  RSMF..... 00       R035..... 0000      TRQI..... 42
+0038 CSCB..... 00F4D048  TSB...... 00B61938  EJST..... 0000001  8C257E00

+0048 EWST..... 9CCDE747  76A09480            JSTL..... 00141A4  ECB...... 808FEF78  UBET..... 9CCDE740
 .
 .
 .
ASSB: 01946600
+0000 ASSB..... ASSB      VAFN..... 00000000  EVST..... 0000000  00000000

+0010 VFAT..... 00000000  00000000            RSV...... 000      XMCC..... 0000      XMCT.....00000000
+0020 VSC...... 00000000  NVSC..... 0000004C  ASRR..... 0000000  R02C..... 00000000  00000000 00000000
+0038           00000000  00000000

*** ADDRESS SPACE SWITCH EVENT MASK OFF (ASTESSEM = 0) ***

TCB: 008D18F0
+0000 RBP...... 008FE7D8  PIE...... 00000000  DEB...... 00B1530  TIO...... 008D4000  CMP......805C6000
+0014 TRN...... 40000000  MSS...... 7FFF7418  PKF...... 80       FLGS..... 01000000  00
+0022 LMP...... FF        DSP...... FE        LLS...... 00D1A88  JLB...... 00011F18  JPQ......00000000
+0030 GPRO-3... 00001000  008A4000  00000000  00000000
+0040 GPR4-7... 00FDC730  008A50C8  00000002  80E73F04
+0050 GPR8-11.. 81CC4360  008A6754  008A67B4  00000008
```

Figure 10. Sample beginning of a SYSUDUMP

Snap dumps

Snap dumps are always sent to the data set defined by the CSQSNAP DD statement. They can be issued by the adapters or the channel initiator.

- Snap dumps are issued by the batch, CICS, IMS, or RRS adapter when an unexpected error is returned by the queue manager for an MQI call. A full dump is produced containing information about the program that caused the problem.

  For a snap dump to be produced, the CSQSNAP DD statement must be in the batch application JCL, CICS JCL, or IMS dependent region JCL.
- Snap dumps are issued by the channel initiator in specific error conditions instead of a system dump. The dump contains information relating to the error. Message CSQX053E is also issued at the same time.

  To produce a snap dump, the CSQSNAP DD statement must be in the channel initiator started-task procedure.

SYS1.LOGREC information

The SYS1.LOGREC data set records various errors that different components of the operating system encounter. For more information about using SYS1.LOGREC records, see the *MVS Diagnosis: Tools and Service Aids* manual.

WebSphere MQ for z/OS recovery routines write information in the *system diagnostic work area* (SDWA) to the SYS1.LOGREC data set when retry is attempted, or when percolation to the next recovery routine occurs. Multiple SYS1.LOGREC entries can be recorded, because two or more retries or percolations might occur for a single error.

The SYS1.LOGREC entries recorded near the time of abend might provide valuable historical information about the events leading up to the abend.

Finding the applicable SYS1.LOGREC information

To obtain a SYS1.LOGREC listing, either:

- Use the EREP service aid, described in the *MVS Diagnosis: Tools and Service Aids* manual to format records in the SYS1.LOGREC data set.
- Specify the VERBEXIT LOGDATA keyword in IPCS.
- Use option 7 on the DUMP ANALYSIS MENU (refer to "Using the WebSphere MQ for z/OS dump display panels" on page 51).

Only records available in storage when the dump was requested are included. Each formatted record follows the heading `*****LOGDATA*****`.

When SVC dumps are not produced

Under some circumstances, SVC dumps are not produced. Generally, dumps are suppressed because of time or space problems, or security violations. The list below summarizes other reasons why SVC dumps might not be produced:

- The z/OS *serviceability level indication processing* (SLIP) commands suppressed the abend.

  The description of IEACMD00 in the *MVS Initialization and Tuning Reference* manual lists the defaults for SLIP commands executed at IPL time.
- The abend reason code was one that does not require a dump to determine the cause of abend.
- SDWACOMU or SDWAEAS (part of the system diagnostic work area, SDWA) was used to suppress the dump.

Suppressing WebSphere MQ for z/OS dumps using z/OS DAE

You can suppress SVC dumps that duplicate previous dumps. The *MVS Diagnosis: Tools and Service Aids* manual gives details about using z/OS *dump analysis and elimination* (DAE).

To support DAE, WebSphere MQ for z/OS defines two *variable recording area* (VRA) keys and a minimum symptom string. The two VRA keys are:

- KEY VRADAE (X'53'). No data is associated with this key.
- KEY VRAMINSC (X'52') DATA (X'08')

WebSphere MQ for z/OS provides the following data for the minimum symptom string in the *system diagnostic work area* (SDWA):

- Load module name
- CSECT name
- Abend code
- Recovery routine name
- Failing instruction area
- REG/PSW difference
- Reason code
- Component identifier
- Component subfunction

Dumps are considered duplicates for the purpose of suppressing duplicate dumps if eight (the X'08' from the VRAMINSC key) of the nine symptoms are the same.

Using trace for problem determination

The trace facilities available with WebSphere MQ for z/OS are:

- The user parameter (or API) trace
- The IBM internal trace used by the support center
- The channel initiator trace
- The line trace

This chapter describes how to collect and interpret the data produced by the user parameter trace, and describes how to produce the IBM internal trace for use by the IBM support center. The other trace facilities that you can use with WebSphere MQ are also discussed.

The user parameter and IBM internal traces

You can obtain information about MQI calls and user parameters passed by some WebSphere MQ calls on entry to, and exit from, WebSphere MQ. To do this, use the global trace in conjunction with the z/OS generalized trace facility (GTF).

Starting the trace

To use the trace for problem determination, you must start the following:

- The GTF for your z/OS system
- The WebSphere MQ trace for each queue manager subsystem for which you want to collect data

Starting the GTF:

When you start the GTF, specify the USRP option. You are prompted to enter a list of event identifiers (EIDs). The EIDs used by WebSphere MQ are:

5E9 To collect information about control blocks on entry to WebSphere MQ

5EA To collect information about control blocks on exit from WebSphere MQ

Sometimes, if an error occurs that you cannot solve yourself, you might be asked by your IBM support center to supply other, internal, trace information for them to analyze. The additional type of trace is:

5EE To collect information internal to WebSphere MQ

You can also use the JOBNAMEP option, specifying the batch, CICS, IMS, or TSO job name, to limit the trace output to specific jobs. Figure 11 illustrates sample startup for the GTF, specifying the four EIDs, and a jobname. The lines shown in bold type **like this** are the commands that you enter at the console; the other lines are prompts and responses.

For more information about starting the GTF trace, see the *MVS Diagnosis: Tools and Service Aids* manual.

```
START GTFxx.yy
 £HASP100 GTFxx.yy  ON STCINRDR
 £HASP373 GTFxx.yy  STARTED
*01 AHL100A SPECIFY TRACE OPTIONS
 R 01,TRACE=JOBNAMEP,USRP
 TRACE=JOBNAMEP,USRP
 IEE600I REPLY TO 01 IS;TRACE=JOBNAMEP,USRP
*02 ALH101A SPECIFY TRACE EVENT KEYWORDS - JOBNAME=,USR=
 R 02,JOBNAME=(xxxxMSTR,xxxxCHIN,zzzzzzzz),USR=(5E9,5EA,5EE)
 JOBNAME=(xxxxMSTR,xxxxCHIN,zzzzzzzz),USR=(5E9,5EA,5EE)
 IEE600I REPLY TO 02 IS;JOBNAME=(xxxxMSTR,xxxxCHIN,zzzzzzzz),USR=(5E9,5EA,5EE)
*03 ALH102A CONTINUE TRACE DEFINITION OR REPLY END
 R 03,END
 END
 IEE600I REPLY TO 03 IS;END
 AHL103I TRACE OPTIONS SELECTED-USR=(5E9,5EA,5EE)
AHL103I JOBNAME=(xxxxMSTR,xxxxCHIN,zzzzzzzz)
*04 AHL125A RESPECIFY TRACE OPTIONS OR REPLY U
 R 04,U
 U
 IEE600I REPLY TO 04 IS;U
 AHL031I GTF INITIALIZATION COMPLETE

where:

     xx is the name of the GTF procedure to use (optional), and yy is an
     identifier for this occurrence of GTF trace.

     xxxx is the name of the queue manager and zzzzzzzz is
     a batch job or CICS region name. Up to 5 job names can be listed.
```

Figure 11. Example startup of GTF to use with the WebSphere MQ trace

When using GTF, specify the primary job name (CHINIT, CICS, or batch) in addition to the queue manager name (xxxxMSTR).

Enabling the trace within WebSphere MQ:

Use the START TRACE command, specifying type GLOBAL to start writing WebSphere MQ records to the GTF. To define the events that you want to produce trace data for, use one or more of the following classes:

CLASS	Event traced
2	Record the MQI call and MQI parameters when a completion code other than MQRC_NONE is detected.
3	Record the MQI call and MQI parameters on entry to and exit from the queue manager.

After the trace has started, you can display information about, alter the properties of, and stop, the trace with the following commands:

- DISPLAY TRACE

- ALTER TRACE
- STOP TRACE

To use any of the trace commands, you must have one of the following:

- Authority to issue start and stop trace commands (trace authority)
- Authority to issue the display trace command (display authority)

Note:

1. The trace commands can also be entered through the initialization input data sets.
2. The trace information produced will also include details of syncpoint flows - for example PREPARE and COMMIT.

For information about these commands, see the WebSphere MQ Script (MQSC) Command Reference manual.

Stopping the GTF:

When you stop the GTF, you must specify the additional identifier (**yy**) used at startup. Figure 12 illustrates a sample stop command for the GTF. The commands shown in bold type **like this** are the commands that you enter at the console.

```
STOP yy
```

Figure 12. Example of GTF Stop command to use with the WebSphere MQ trace

Formatting the information

To format the user parameter data collected by the global trace, use either the batch job shown in Figure 13 on page 68 or the IPCS `GTFTRACE USR(xxx)` command, where *xxx* is:

5E9 To format information about control blocks on entry to WebSphere MQ MQI calls

5EA To format information about control blocks on exit from WebSphere MQ MQI calls

5EE To format information about WebSphere MQ internals

You can also specify the `JOBNAME(jobname)` parameter to limit the formatted output to specific jobs.

```
//S1 EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=4096K
//IPCSPARM DD DSN=SYS1.PARMLIB,DISP=SHR
//IPCSDDIR DD DSN=thlqual.ipcs.dataset.directory,DISP=SHR
//SYSTSPRT DD SYSOUT=*,DCB=(LRECL=137)
//IPCSTOC  DD SYSOUT=*
//GTFIN  DD  DSN=gtf.trace,DISP=SHR
//SYSTSIN DD *
IPCS
SETDEF FILE(GTFIN) NOCONFIRM
GTFTRACE USR(5E9,5EA,5EE)
/*
//STEPLIB  DD  DSN=thlqual.SCSQAUTH,DISP=SHR
```

Figure 13. Formatting the GTF output in batch. thlqual is your high level qualifier for WebSphere MQ data sets, and gtf.trace is the name of the data set containing your trace information. You must also specify your IPCS data set directory.

Identifying the control blocks associated with WebSphere MQ:

The format identifier for the WebSphere MQ trace is D9. This value appears at the beginning of each formatted control block in the formatted GTF output, in the form:

```
USRD9
```

Identifying the event identifier associated with the control block:

The trace formatter inserts one of the following messages at the top of each control block. These indicate whether the data was captured on entry to or exit from WebSphere MQ

- CSQW072I ENTRY: MQ user parameter trace
- CSQW073I EXIT: MQ user parameter trace

If trace data is not produced

If trace data is not produced, check the following:

- Was the GTF started correctly, specifying EIDs 5E9, 5EA, and 5EE on the USRP option?
- Was the START TRACE(GLOBAL) command entered correctly, and were the relevant classes specified?

Interpreting the information

When you look at the data produced by the GTFTRACE command, consider the following points:

- If the control block consists completely of zeros, it is possible that an error occurred while copying data from the user's address space. This could be because an invalid address was passed.
- If the first part of the control block contains non-null data, but the rest consists of zeros, it is again possible that an error occurred while copying data from the user's address space, for example, the control block was not placed entirely within valid storage. This could also be due to the control block not being initialized correctly.
- If the error has occurred on exit from WebSphere MQ, it is possible that WebSphere MQ could not write the data to the user's address space. The data displayed is the version that it was attempting to copy to the user's address space.

The control blocks traced:

Table 9 illustrates which control blocks are traced for different MQI calls.

Table 9. Control blocks traced for WebSphere MQ MQI calls

MQI call	Entry	Exit
MQOPEN	MQOD	MQOD
MQCLOSE	None	None
MQPUT	MQMD, MQPMO, and the first 256 bytes of message data	MQMD, MQPMO, and the first 256 bytes of message data
MQPUT1	MQMD, MQOD, MQPMO, and the first 256 bytes of message data	MQMD, MQOD, MQPMO, and the first 256 bytes of message data
MQGET	MQMD, MQGMO	MQMD, MQGMO, and the first 256 bytes of message data
MQINQ	Selectors (if *SelectorCount* is greater than 0)	Selectors (if *SelectorCount* is greater than 0) Integer attributes (if *IntAttrCount* is greater than 0) Character attributes (if *CharAttrLength* is greater than 0)
MQSET	Selectors (if *SelectorCount* is greater than 0) Integer attributes (if *IntAttrCount* is greater than 0) Character attributes (if *CharAttrLength* is greater than 0)	Selectors (if *SelectorCount* is greater than 0) Integer attributes (if *IntAttrCount* is greater than 0) Character attributes (if *CharAttrLength* is greater than 0)

Note: In the special case of an **MQGET** call with the WAIT option, a double entry is seen if there is no message available at the time of the **MQGET** request, but a message subsequently becomes available before the expiry of any time interval specified.

This is because, although the application has issued a single **MQGET** call, the adapter is performing the wait on behalf of the application and when a message becomes available it reissues the call. So in the trace it will appear as a second **MQGET** call.

Information about specific fields of the queue request parameter list is also produced in some circumstances. The fields in this list are identified as follows:

Identifier	Description
BufferL	Buffer length
CompCode	Completion code
CharAttL	Character attributes length
DataL	Data length
Hobj	Object handle
IntAttC	Count of integer attributes
pObjDesc	Object descriptor

Identifier	Description
Options	Options
pBuffer	Address of buffer
pCharAtt	Address of character attributes
pECB	Address of ECB used in get
pGMO	Address of get message options
pIntAtt	Address of integer attributes
pMsgDesc	Address of message descriptor
pPMO	Address of put message options
pSelect	Address of selectors
Reason	Reason code
RSVn	Reserved for IBM
SelectC	Selector count
Thread	Thread
UOWInfo	Information about the unit of work
Userid	CICS or IMS user ID, for batch or TSO this is zero

The Log and Trace Analyzer tool

The Log and Trace Analyzer tool is an interface that allows you to work with logs and traces produced by different components of a deployed system. It provides a single point of contact for importing and analyzing log files or trace files from multiple products and allows you to determine the relationship between the events captured by these products (correlation).

You can download the tool from the Log and Trace Analyzer tool download site at
`http://www.ibm.com/developerworks/autonomic/btmpd/`

You can find online help for the Log and Trace Analyzer tool in the Autonomic Computing Toolkit User's Guide at:
`http://download.boulder.ibm.com/ibmdl/pub/software/dw/autonomic/books/fpu3mst.htm`

Examples of trace output

Figure 14 on page 71 shows an example of a trace taken on entry to an MQPUT1 call. The following items have been produced:

- Queue request parameter list
- Object descriptor (MQOD)
- Message descriptor (MQMD)
- Put message options (MQPMO)
- The first 256 bytes of message data

Compare this to Figure 15 on page 72, which illustrates the same control blocks on exit from WebSphere MQ.

```
USRD9 5E9 ASCB 00F87E80           JOBN ECIC330
CSQW072I ENTRY: MQ user parameter trace
PUTONE
    Thread... 004C2B10  Userid... CICSUSER  pObjDesc. 106B2010
    pMsgDesc. 106B20B8  pPMO..... 106B2200
    BufferL.. 00000064  pBuffer.. 106A0578  RSV1..... 00000000
    RSV2..... 00000000  RSV3..... 116BC830
    C9E8C1E8  C5C3C9C3  AA8E8583  76270484  | IYAYECIC..ec...d |
    D4D8E3E3  0000048C  00000000  00000000  | MQTT............ |
    00000000  1910C7C2  C9C2D4C9  E8C14BC9  | ......GBIBMIYA.I |
    C7C3E2F2  F0F48E85  83762979  00010000  | GCS204.ec..`.... |

              GMT-01/30/05 14:42:08.412320    LOC-01/30/05 14:42:08.412320

USRD9 5E9 ASCB 00F87E80           JOBN ECIC330
CSQW072I ENTRY: MQ user parameter trace
 +0000  D6C44040  00000001  00000000  C2404040  | OD  ........B    |
 +0010  40404040  40404040  40404040  40404040  |                  |
 ...
 +00A0  00000000  00000000                     | ........         |

              GMT-01/30/05 14:42:08.412345    LOC-01/30/05 14:42:08.412345

USRD9 5E9 ASCB 00F87E80           JOBN ECIC330
CSQW072I ENTRY: MQ user parameter trace
 +0000  D4C44040  00000001  00000000  00000008  | MD  ............ |
 ...
 +0130  40404040  40404040  40404040  40404040  |                  |
 +0140  40404040                               |                  |

              GMT-01/30/05 14:42:08.412370    LOC-01/30/05 14:42:08.412370

USRD9 5E9 ASCB 00F87E80           JOBN ECIC330
CSQW072I ENTRY: MQ user parameter trace
 +0000  D7D4D640  00000001  00000000  FFFFFFFF  | PMO ............ |
 ...
 +0070  40404040  40404040  40404040  40404040  |                  |

              GMT-01/30/05 14:42:08.412393    LOC-01/30/05 14:42:08.412393

USRD9 5E9 ASCB 00F87E80           JOBN ECIC330
CSQW072I ENTRY: MQ user parameter trace
 +0000  C1C1C1C1  C1C1C1C1  C1404040  40404040  | AAAAAAAAA        |
 ...
 +0060  40404040                               |                  |

              GMT-01/30/05 14:42:08.412625    LOC-01/30/05 14:42:08.412625
```

Figure 14. Example trace data from an entry trace of an MQPUT1 request

```
USRD9 5EA ASCB 00F87E80            JOBN ECIC330
CSQW073I EXIT:  MQ user parameter trace
PUTONE
    Thread... 004C2B10  Userid... CICSUSER  pObjDesc. 106B2010
    pMsgDesc. 106B20B8  pPMO..... 106B2200
    BufferL.. 00000064  pBuffer.. 106A0578  RSV1..... 00000000
    RSV2..... 00000000  RSV3..... 116BC830
    CompCode. 00000002  Reason... 000007FB
    C9E8C1E8  C5C3C9C3  AA8E8583  76270484  | IYAYECIC..ec...d |
    D4D8E3E3  0000048C  00000000  00000000  | MQTT............ |
    00000000  1910C7C2  C9C2D4C9  E8C14BC9  | ......GBIBMIYA.I |
    C7C3E2F2  F0F48E85  83762979  00010000  | GCS204.ec..`.... |
MQRC_OBJECT_TYPE_ERROR

               GMT-01/30/05 14:42:08.412678    LOC-01/30/05 14:42:08.412678

USRD9 5EA ASCB 00F87E80            JOBN ECIC330
CSQW073I EXIT:  MQ user parameter trace
 +0000  D6C44040  00000001  00000000  C2404040  | OD  ........B    |
 ...
 +00A0  00000000  00000000                     | ........         |

               GMT-01/30/05 14:42:08.412789    LOC-01/30/05 14:42:08.412789

USRD9 5EA ASCB 00F87E80            JOBN ECIC330
CSQW073I EXIT:  MQ user parameter trace
 +0000  D4C44040  00000001  00000000  00000008  | MD  ............ |
 ...
 +0140  40404040                               |                  |

               GMT-01/30/05 14:42:08.412814    LOC-01/30/05 14:42:08.412814

USRD9 5EA ASCB 00F87E80            JOBN ECIC330
CSQW073I EXIT:  MQ user parameter trace
 +0000  D7D4D640  00000001  00000000  FFFFFFFF  | PMO ............ |
 ...
 +0070  40404040  40404040  40404040  40404040  |                  |

               GMT-01/30/05 14:42:08.412836    LOC-01/30/05 14:42:08.412836

USRD9 5EA ASCB 00F87E80            JOBN ECIC330
CSQW073I EXIT:  MQ user parameter trace
 +0000  C1C1C1C1  C1C1C1C1  C1404040  40404040  | AAAAAAAAA        |
 ...
 +0060  40404040                               |                  |

               GMT-01/30/05 14:42:08.412858    LOC-01/30/05 14:42:08.412858
```

Figure 15. Example trace data from an exit trace of an MQPUT1 request

Other types of trace

You might also find it helpful to use the following trace facilities with WebSphere MQ.

The channel initiator trace

See Figure 4 on page 50 for information about how to get a dump of the channel initiator address space. Note that dumps produced by the channel initiator do not

include trace data space. The trace data space, which is called CSQXTRDS, contains trace information. You can request this by specifying it on a slip trap or when you use the dump command.

To run the trace using the START TRACE command, see the WebSphere MQ Script (MQSC) Command Reference manual. You can set this trace to start automatically using the TRAXSTR queue manager attribute. For more information about how to do this, see the WebSphere MQ Script (MQSC) Command Reference.

You can display this trace information by entering the IPCS command:

```
LIST 1000. DSPNAME(CSQXTRDS)
```

You can format it using the command:

```
CTRACE COMP(CSQXssnm)
```

where *ssnm* is the subsystem name.

The line trace

A wrap-around line trace exists for each channel. This trace is kept in a 4 KB buffer for each channel in the channel initiator address space. Trace is produced for each channel, so it is ideal for problems where a channel appears to be hung, because information can be collected about the activity of this channel long after the normal trace has wrapped.

The line trace is always active; you cannot turn it off. It is available for both LU 6.2 and TCP channels and should reduce the number of times a communications trace is required.

You can view the trace as unformatted trace that is written to CSQSNAP. You can display the trace by following these steps:

1. Ensure that the CHIN procedure has a SNAP DD statement.
2. Start a CHIN trace, specifying IFCID 202 as follows:

   ```
   START TRACE(CHINIT) CLASS(4) IFCID(202)
   ```
3. Display the channel status for those channels for which the line trace is required:

   ```
   DISPLAY CHSTATUS(channel) SAVED
   ```

   This dumps the current line for the selected channels to CSQSNAP. See “Snap dumps” on page 63 for further information.

   Note:

   a. The addresses of the storage dump are incorrect because the CSQXFFST mechanism takes a copy of the storage before writing it to CSQSNAP.

   b. The dump to CSQSNAP is only produced the first time you run the DISPLAY CHSTATUS SAVED command. This is to prevent getting dumps each time you run the command.

      To obtain another dump of line trace data, you must stop and restart the current trace.

      1) You can use a selective STOP TRACE command to stop just the trace that was started to gather the line trace data. To do this, note the TRACE NUMBER assigned to the trace as shown in the example below:

         ```
         +ssid START TRACE(CHINIT) CLASS(4) IFCID(202)
              CSQW130I +ssid 'CHINIT' TRACE STARTED, ASSIGNED TRACE NUMBER 01
         ```

2) To stop the trace, issue the following command:
   ```
   +ssid STOP TRACE(CHINIT) TNO(01)
   ```
3) You can then enter another START TRACE command with a DISPLAY CHSTATUS SAVED command to gather more line trace data to CSQSNAP.

4. The line trace buffer is unformatted. Each entry starts with a clock, followed by a time stamp, and an indication of whether this is an OUTBOUND or INBOUND flow. Use the time stamp information to find the earliest entry.

The CICS adapter trace

The CICS adapter writes entries to the CICS trace if your trace number is set to a value in the range 0 through 199 (decimal), and if either:

- CICS user tracing is enabled, or
- CICS internal/auxiliary trace is enabled

You can enable CICS tracing in one of two ways:

- Dynamically, using the CICS-supplied transaction CETR
- By ensuring that the USERTR parameter in the CICS system initialization table (SIT) is set to YES

For more information about enabling CICS trace, see the *CICS Problem Determination Guide*.

The CICS trace entry originating from the CICS adapter has a value AP0*000*, where *000* is the hexadecimal equivalent of the decimal value of the CICS adapter trace number you specified.

The trace entries are shown in Chapter 8, “CICS adapter trace entries,” on page 103.

System SSL trace

You can collect System SSL trace using the SSL Started Task. The details of how to set up this task are in the *System Secure Sockets Layer Programming* book, SC24-5901. A trace file is generated for each SSLTASK running in the CHINIT address space.

z/OS traces

z/OS traces, which are common to all products operating as formal subsystems of z/OS, are available for use with WebSphere MQ. For information about using and interpreting this trace facility, see the *MVS Diagnosis: Tools and Service Aids* manual.

Chapter 4. Finding solutions to similar problems

Searching the IBM database

IBM keeps records of all known problems with its licensed programs on its software support database (RETAIN). IBM support center staff continually update this database as new problems are found, and they regularly search the database to see if problems they are told about are already known.

If you have access to one of IBM's search tools such as INFORMATION/ACCESS OR INFORMATION/SYSTEM you can look on the RETAIN database yourself. If not, you can contact the IBM support center to perform the search for you.

You can search the database using a string of keywords to see if a similar problem already exists. This section explains how to search the database using keywords.

You can use the keyword string (also called the symptom string) that appears in a dump or SYS1.LOGREC record to search the database, or you can build your own keyword string from the procedure described in "Building a keyword string" on page 78. Before you use the procedures in this section, work through "Preliminary checks" on page 2 to check that the problem does not have a simple solution.

If the search is successful, you find a similar problem description and, usually, a fix. If the search is unsuccessful, you should use these keywords when contacting IBM for additional assistance, or when documenting a possible *authorized program analysis report* (APAR).

Searching the IBM software support database is most effective if you:

- Always spell keywords the way they are spelled in this book
- Include all the appropriate keywords in any discussion with your IBM support center

Search argument process

Use the following procedure when searching the IBM software support database:

1. Using INFORMATION/ACCESS or INFORMATION/SYSTEM, search the database using the keywords you have developed. Details on how to construct suitable keywords are given in "Building a keyword string" on page 78

   Note: Do *not* use both the CSECT keyword and the load module modifier keyword at the same time for the first search. Refer to "Load module modifier keyword" on page 84 for additional information.
2. Compare each matching APAR closing description with the current failure symptoms.
3. If you find an appropriate APAR, apply the correction or PTF.
4. If you do not find an appropriate APAR, vary the search argument by following the suggestions provided under "Techniques for varying the search" on page 76.
5. If you still cannot find a similar problem, see "Resolving a problem" on page 97.

Techniques for varying the search

To vary your search, follow these guidelines:

Dropping keywords to widen your search

If you used a complete set of keywords (as described in "Building a keyword string" on page 78) and could not find any problem descriptions to examine, drop one or more of the following keywords and try again:

- Release-level keyword
- Load Module modifier keyword
- Recovery routine modifier keyword
- CSECT keyword

Adding keywords to narrow your search

If you tried to search with an incomplete set of keywords and found too many problem descriptions to examine, add keywords to narrow your search. For example, for storage manager abends (which produce a reason code beginning with X'00E2'), you use the CSECT name recorded in the VRA to narrow or vary the search.

Making your set of keywords more precise

If you tried to search with a complete set of keywords and found too many matching descriptions *and* if you received a 4-byte WebSphere MQ abend reason code, you might be able to make your set of keywords more precise. Look up the 4-byte abend reason code in the WebSphere MQ for z/OS Messages and Codes manual to find additional information available for this problem.

Replacing keywords to locate problems

If your type-of-failure keyword is WAIT, LOOP, or PERFM, and if you did not find a matching problem description, replace that keyword with one of the other two listed here. Sometimes a problem that appears to be a performance problem might actually be a WAIT or LOOP; likewise, a problem that seems to be a WAIT or a LOOP might actually be recorded as a performance problem.

Using message numbers in your search

If your type-of-failure keyword is MSGx and you received more than one message near the time of the problem, repeat the search replacing the message number in the keyword with the number of each related message in turn.

Using DOC as a keyword in your search

If your type-of-failure keyword is MSGx, PERFM, or INCORROUT, and if the problem occurred immediately after you performed some action that a WebSphere MQ book told you to perform, the problem could be recorded as a DOC type of failure. In this case, try searching with DOC as your type-of-failure keyword, rather than with MSGx, PERFM, or INCORROUT.

The keyword format

The keywords in "Building a keyword string" on page 78 are described in two distinct formats: the z/OS, or free format; and the structured database (SDB) format. Structured symptoms are also called RETAIN symptoms and "failure keywords".

If your installation has a tool for performing structured searches, you can use the SDB format. Otherwise, you should use the free format. For both formats, your choice of keywords depends on the type of failure that occurred.

Free format

A free-form keyword can consist of any piece of data that is related to the problem. To help you search the database, a set of keywords has been defined, and you can use them to narrow your search. (For example, if you know the name of the CSECT in error, you can use this to search, but if you add the MSGxx or ABEND keyword, your search will be more precise.)

The following list shows keywords defined for use in a free format search:

Table 10. Keywords defined for use in a free format search

Keyword	Meaning
ABEND	Abnormal termination of a task; no error message.
ABENDxx	Abnormal termination of a task; xx is the abend code.
ABENDUxx	User abend; xx is the abend code.
DOC	Documentation discrepancy that caused a problem.
HALTxx	Halt; xx is the halt number.
INCORROUT	Any incorrect data output, except performance degradation.
INTEG	Integrity problem.
LOOP	Loop.
MSGxx	Any message; xx is the message identifier.
PERFM	Performance degradation.
PROCCHK	Processor check.
PROGCH	Program check.
WAIT	Wait condition; undocumented and no identifier.
WAITxx	System wait condition; xx is the identifier.

Structured database (SDB) format

The structured symptoms consist of a prefix keyword, which identifies the type of symptom, followed by a slash (/) and the data portion of the symptom.

- The prefix keyword has one through eight characters.
- All characters must be alphanumeric, #, @, or $.
- At least one character of data is required.
- The maximum length, including the prefix, is 15 characters.

For example, the following is a structured symptom string for a message identifier of CSQC223D:

```
MS/CSQC223D
```

The following list shows the structured symptom strings:

Table 11. Keywords defined for use in a free format search

Keyword	Meaning
AB	Abend code.

Table 11. Keywords defined for use in a free format search (continued)

Keyword	Meaning
FLDS	Name of a field or control block involved with the problem.
LVLS	Level of the base system or licensed program.
MS	Message identifier.
OPCS	Operation code (opcode) for software, such as an assembler-language opcode.
PCSS	Program command or other software statement, such as JCL, a parameter, or a data set name.
PIDS	Program identifier for a component involved in the problem.
PRCS	Program return code, generated by software, including reason codes and condition codes.
PTFS	Program temporary fix (PTF) for software associated with a problem.
PUBS	Identifier of a publication associated with a problem.
RECS	Record associated with a problem.
REGS	Register for a software program associated with a problem. The value can be the register/PSW difference (*rrddd*), which the STATUS FAILDATA subcommand of IPCS provides for abends. The difference (*ddd*) is a hexadecimal offset from a probable base register or branch register (*rr*).
RIDS	Routine identifier, such as the name of a CSECT or subroutine. If the RIDS/ value has no suffix, the value is a CSECT name. The following suffixes are supported: • #L — for a load module • #R — for a recovery routine
VALU	Value in a field or register. One of the following qualifiers is required as the first character of the value: • B — for a bit value • C — for a character value • H — for a hexadecimal value
WS	Wait state code issued by the system, or device-issued wait code. One of the following qualifiers is required as the first character of the value: • D — for disabled wait (system disabled for I/O or external interrupts) • E — for enabled wait

For more information about which prefix keyword to use for which type of symptom, see Chapter 6, “SDB format symptom-to-keyword cross reference,” on page 99.

Building a keyword string

This chapter describes a systematic way of selecting *keywords* to describe a failure in WebSphere MQ for z/OS. Keywords are predefined words or abbreviations that identify aspects of a program failure.

To determine which WebSphere MQ for z/OS keywords to use and the procedures for selecting them, refer to the flowchart in Figure 16 on page 80.

To begin selecting your keywords:

1. Follow the procedures in "Component-identifier keyword" on page 81 and "Release-level keyword" on page 81. Do this for all failures.
2. Follow one of the type-of-failure keyword procedures.
3. Identify the area of the failure using CSECT and modifier keywords when appropriate. The procedures in this section refer you to these steps as needed.
4. Follow "Searching the IBM database" on page 75 to learn how to search the database with your set of keywords. Do this for all failures.
5. If the search is unsuccessful, turn to "Resolving a problem" on page 97. This helps IBM product support personnel determine whether an APAR should be submitted.

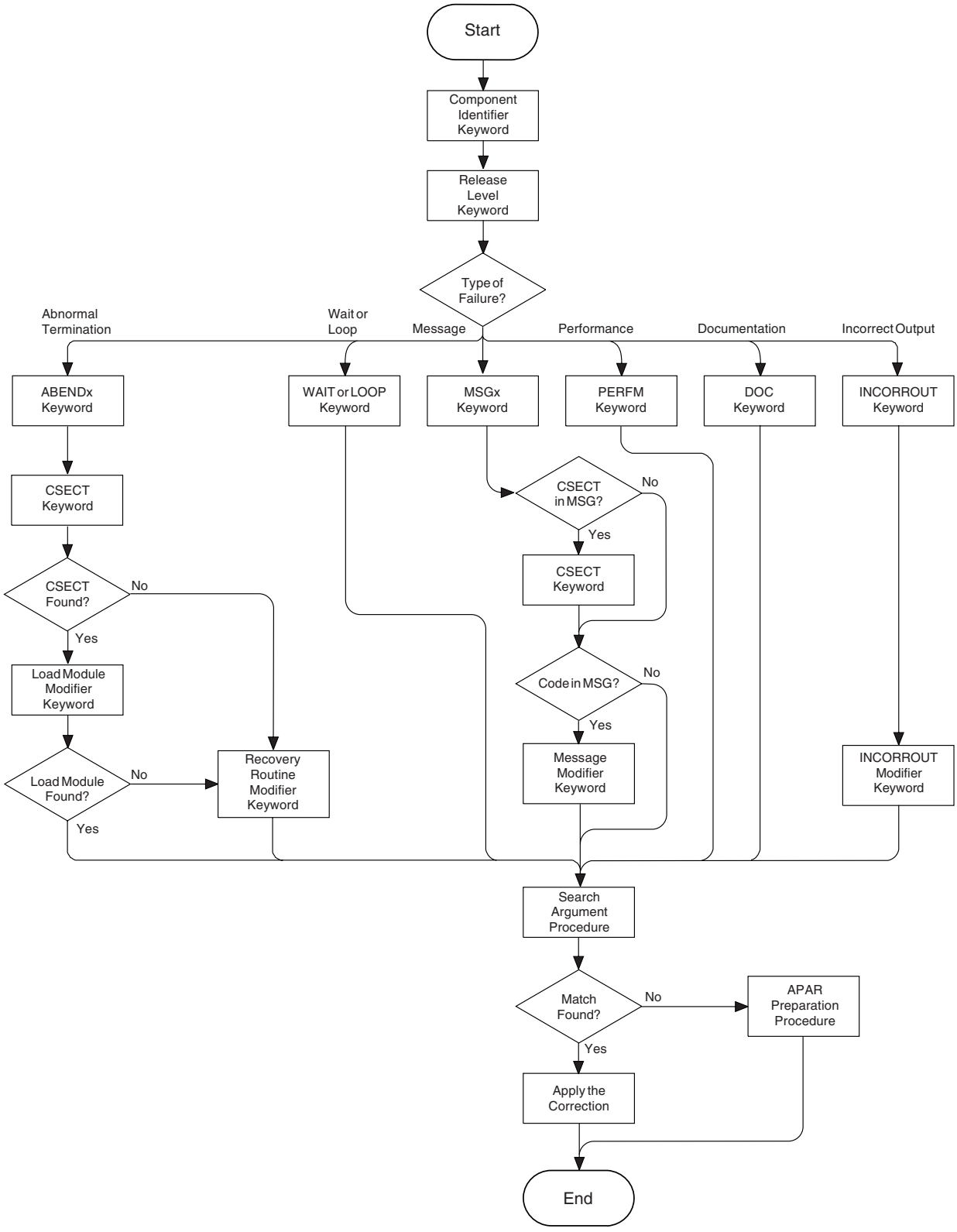

Figure 16. High-level flowchart of various sets of keywords

Component-identifier keyword

The *component-identifier keyword* identifies the library within the IBM software support database that contains *authorized program analysis reports* (APARs) and *program temporary fixes* (PTFs) for the product.

The component-identifier keyword for WebSphere MQ for z/OS is **5655R3600**.

This section describes how to determine the nine-digit component identifier keyword for your failure to verify that the problem was caused by WebSphere MQ for z/OS. If the component identifier is not 5655R3600, the problem could be caused by another product.

If the problem caused a dump, display the dump title, locate the COMP= label, and note the first five characters following that label. If these characters are **R3600**, the problem was caused by WebSphere MQ for z/OS. Append those five characters to **5655** and use this as the first keyword in your search argument.

```
ssnm,ABN=compltn-reason,U=userid,C=compid.release.comp-function,
M=module, LOC=loadmod.csect+csect_offset
```

If you cannot use the dump title, display the z/OS SYMPTOM STRING in the formatted dump. Note the nine characters following the PIDS/ label.

Release-level keyword

The *release-level keyword* narrows the symptom search to your specific release level. Using this keyword is optional, but recommended, when searching the IBM software support database. It is required, however, when an APAR is submitted.

Locate the three-digit release identifier in the dump title. It follows COMP=R3600, for example:

```
COMP=L8200.700
```

Add this to your keyword string, in one of the formats shown below:

Free format

```
5655R3600 R700
```

Structured format

```
PIDS/5655R3600 LVLS/700
```

Type-of-failure keyword

To narrow your search, use one or more of the type-of-failure and modifier keywords to describe an external symptom of a program failure. The various types of failures are shown in Table 12. Use this table to find the name and page number of the keyword that best matches your problem.

Table 12. Types of WebSphere MQ for z/OS failures

Problem	Procedure
Abend of the subsystem or task	"Abend keyword" on page 82
Unexpected program suspension	"Wait and loop keywords" on page 85
Uncontrolled program looping (often signaled by repeating messages or output)	"Wait and loop keywords" on page 85

Table 12. Types of WebSphere MQ for z/OS failures (continued)

Problem	Procedure
Errors signaled by or associated with messages	"Message keyword" on page 85
Performance degradation	"Performance keyword" on page 87
Documentation problem	"Documentation keyword" on page 88
Unexpected or missing output	"Incorrect output keyword" on page 89

Abend keyword

Use the ABEND keyword when the subsystem or task terminates abnormally. This procedure describes how to locate the abend completion code and the abend reason code (if there is one), and how to use them in a set of keywords. Check the SYS1.LOGREC to determine how many abends there were. Sometimes an earlier abend causes a secondary abend that causes a dump. If no dump has been taken, try searching the database with a minimum symptom string (the component-identifier, and release-level keywords). If you cannot find any information that seems to relate to your problem, contact your IBM support center.

When a WebSphere MQ for z/OS abend occurs, you will see one of the following symptoms:

- An IEA911E message from z/OS, indicating that an SVC dump occurred. See "IEA911E message."
- The CSQV086E message QUEUE MANAGER ABNORMAL TERMINATION REASON=xxxxxxxx. See "CSQV086E message" on page 83.

IEA911E message

1. Use the `DISPLAY DUMP,TITLE` command on the console to display the SVC dump title for this abend, or use one of the methods described in "WebSphere MQ dumps" on page 48 to look at the dump title in the dump.

   Note: If the first five characters of the COMP field are not R3600, or the dump title is not of the same form as Figure 8 on page 61 or Figure 9 on page 62, the problem was not caused by WebSphere MQ for z/OS, or you are looking at the wrong dump.
2. Locate the 3-character completion code following the word ABND.
   - If the completion code is X'071', or X'122', the operator pressed the RESTART key or canceled the job, probably to break a loop. Verify that this is the case, and turn to "Wait and loop keywords" on page 85.
   - Otherwise, add this to your keyword string, in one of the formats shown below (in this example, X'0C4' is used):

     Free format

     ```
     5655R3600 R700 ABEND0C4
     ```

     Structured format

     ```
     PIDS/5655R3600 LVLS/700 AB/S00C4
     ```
3. Some abends also have reason codes. These reason codes are usually found in message CSQV086E, and register 15 at the time of the abend. Locate the reason code for the abend either:
   - In the 4-byte reason code field in a dump title generated by WebSphere MQ for z/OS

- In the registers at time of error in the abstract information section of the dump
- From the value of register 15 in the error summary display

4. If the completion code is X'5C6', review the diagnostic information for the reason code in the WebSphere MQ for z/OS Messages and Codes manual. Follow any procedures recommended there.

   If the completion code is anything else, and you have found a reason code, check the value against the description of the abend code in the *MVS System Codes* manual to see if it is valid for the abend completion code.

5. Add the reason code to the keyword string (in this example X'00E20015' is used):

   Free format

   ```
   5655R3600 R700 ABEND5C6 RC00E20015
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 AB/S05C6 PRCS/00E20015
   ```

   Turn to "CSECT keyword" on page 84.

CSQV086E message

1. Issue the DISPLAY DUMP command to see whether any SVC dumps occurred near the time the message appeared. (See the *MVS System Commands* manual if necessary.)
2. If there was only one SVC dump for the abend, follow the procedure starting at step 1 on page 82.
3. If there were two or more SVC dumps, follow the steps below.
   a. Read the sections in the WebSphere MQ for z/OS Messages and Codes manual that describe the reason code appearing in your message, and any reason codes appearing in the SVC dump titles. Reason codes appear after the completion code in the SVC dump title. For an example, see "Analyzing the dump" on page 61.
   b. Compare the reason codes in the SVC dumps to determine which dump relates to the CSQV086E message.
   c. Use that SVC dump and follow the procedure starting at step 1 on page 82.
4. If there were two or more *different* abends, follow the steps below:
   a. Determine which abend was the original cause by reviewing the time stamps in the SYS1.LOGREC entries.
   b. Use that SVC dump and follow the procedure starting at step 1 on page 82.
5. If there were no SVC dumps for the abend, follow the steps below.
   a. Locate the 4-byte reason code in the message.
   b. Review the diagnostic information in the WebSphere MQ for z/OS Messages and Codes manual. Follow any procedures recommended there.
   c. Add this to your keyword string, in one of the formats shown below (in this example, a reason code of X'00D93001' is used):

      Free format

      ```
      5655R3600 R700 ABEND6C6 RC00D93001
      ```

      Structured format

      ```
      PIDS/5655R3600 LVLS/700 AB/S06C6 PRCS/00D93001
      ```

      Turn to "CSECT keyword" on page 84.

CSECT keyword

To find the name of the failing CSECT, locate the LOC= label; the second word following it is the CSECT name. For an example, see "Analyzing the dump" on page 61.

Any CSECT name you locate should begin with the letters CSQ or CMQ. If you find a CSECT name with a different prefix, the problem is probably not in WebSphere MQ for z/OS.

Add the CSECT name to your keyword string:

Free format

5655R3600 R700 ABEND0C4 **CSQVATRM**

Structured format

PIDS/5655R3600 LVLS/700 AB/S00C4 **RIDS/CSQVATRM**

If required, narrow your search further by referring to "Load module modifier keyword."

If you cannot find the CSECT, turn to "Recovery routine modifier keyword" on page 85.

Load module modifier keyword

Use the load module modifier keyword to identify the name of the load module involved if your search using the CSECT keyword was unsuccessful, or yielded too many possible matches:

- If your search was unsuccessful, replace the CSECT name with the load module name and try again.
- If your search yielded too many possible matches, add the load module name to your string to further narrow the search.

All WebSphere MQ for z/OS load module names begin with CSQ or CMQ. If you follow these instructions and find a load module name with a different prefix, the problem is in another product.

To locate the load module name, locate the first word following the label LOC=. This is the load module name, and it precedes the CSECT name. (For an example, see "Analyzing the dump" on page 61.)

Add the load module name to your keyword string, or substitute it for the CSECT name as appropriate. If you are using the structured format, follow the name of the module with the characters #L to indicate that this is a load module. Search the database again using the revised keyword string. (See "Searching the IBM database" on page 75.)

Free format

5655R3600 R700 ABEND5C6 RC00E50013 **CSQSLD1** CSQSVSTK (with load module name and then CSECT name)

5655R3600 R700 ABEND5C6 RC00E50013 **CSQSLD1** (with load module name only)

Structured format

PIDS/5655R3600 LVLS/700 AB/S05C6 PRCS/00E50013 **RIDS/CSQSLD1#L** RIDS/CSQSVSTK (with load module name and then CSECT name)

PIDS/5655R3600 LVLS/700 AB/S05C6 PRCS/00E50013 **RIDS/CSQSLD1#L** (with load module name only)

Recovery routine modifier keyword

Include the name of the recovery routine only when you could not determine the names of the CSECT and load module involved at the time of failure, after looking in both the SVC dump and the SYS1.LOGREC entry.

To obtain the recovery routine name, locate the area of the dump title containing the symbol M=. The word following this identifies the functional recovery routine (FRR) or the extended specify task abnormal exit (ESTAE). For an example, see "Analyzing the dump" on page 61.

Add this word to your keyword string. If you are using the structured format, follow the name of the module with the characters #R to indicate that this is a recovery routine. Search the database (see "Searching the IBM database" on page 75).

Free format

```
5655R3600 R700 ABEND5C6 RC00E20015 CSQTFRCV
```

Structured format

```
PIDS/5655R3600 LVLS/700 AB/S05C6 PRCS/00E20015 RIDS/CSQTFRCV#R
```

Wait and loop keywords

If the problem occurred immediately after you did something a WebSphere MQ manual told you to do, the problem might be related to the manual. If you think that this is the case, turn to "Documentation keyword" on page 88.

If you have verified that the wait or loop problem cannot be resolved through other means, use the following procedure:

1. Add WAIT or LOOP to your keyword string, in one of the formats shown below (in this example **WAIT** is used).

   Free format

   ```
   5655r3600 R700 WAIT
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 WAIT
   ```

2. Turn to "Searching the IBM database" on page 75.

Message keyword

Use the MSG keyword if an error is associated with a WebSphere MQ for z/OS message. If you received multiple messages for one error, search the database using the first message issued. If unsuccessful, search the database using the next message, then the next, and so on.

To see if other messages related to your problem have been issued, check the console for WebSphere MQ for z/OS messages, as well as messages issued by other products. If any message is prefixed with "IEC", indicating it was issued by data management services, check the SYSLOG for messages that identify associated data set problems. SYSLOG can also help to diagnose user errors.

If your message was issued immediately after you did something that a WebSphere MQ manual told you to do, the problem might be related to the documentation rather than to the message. If this is the case, turn to "Documentation keyword" on page 88. Otherwise, compare the message prefix with those shown in the table below to determine the appropriate procedure to follow.

Table 13. Message prefixes

Prefix	Component	Procedure
AMQ	WebSphere MQ (not z/OS)	Consult WebSphere MQ Messages
AMT	WebSphere MQ	Consult WebSphere MQ Programmable Command Formats and Administration Interface
ATB	APPC	Consult *MVS System Messages*
ATR	Resource recovery services	Consult *MVS System Messages*
CBC	C/C++	Consult *C/MVS™ User's Guide*
CEE	Language Environment®	Consult *Language Environment Debugging Guide and Run-Time Messages*
CSQ	WebSphere MQ for z/OS	Follow "Procedure for WebSphere MQ for z/OS messages" on page 87
CSV	Contents supervision	Consult *MVS System Messages*
DFH	CICS	Consult *CICS Messages and Codes*
DFS™	IMS	Consult *IMS/ESA Messages and Codes*
DSN	DB2	Consult *DB2 Messages and Codes*
EDC	Language Environment	Consult *Language Environment Debugging Guide and Run-Time Messages*
EZA, EZB, EZY	TCP/IP	Consult *z/OS V2R6.0 eNetwork CS IP Messages: Volumes 1, 2, and 3*
IBM	Language Environment	Consult *Language Environment Debugging Guide and Run-Time Messages*
ICH	RACF	Consult *z/OS Security Server (RACF) Messages and Codes*
IDC	Access method services	Consult *MVS System Messages*
IEA	z/OS system services	Consult *MVS System Messages*
IEC	Data management services	Consult *MVS System Messages*
IEE, IEF	z/OS system services	Consult *MVS System Messages*
IKJ	TSO	Consult *MVS System Messages*
IST	VTAM®	Consult *VTAM Messages*
IWM	MVS workload management services	Consult *MVS System Messages*
IXC	Cross-system Coupling Facility (XCF)	Consult *MVS System Messages*
IXL	Cross-system Extended Services® (XES)	Consult *MVS System Messages*

Procedure for WebSphere MQ for z/OS messages

1. Check whether the name of the CSECT issuing the message appears. This name follows the message number. If no CSECT name appears, only one CSECT can issue this message.
2. Determine whether the message contains any variables, such as return or reason codes.
3. If no CSECT name appears, add the message number to your keyword string, in one of the formats shown below (in this example, message CSQJ006I is used):

   Free format

   ```
   5655R3600 R700 MSGCSQJ006I
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 MS/CSQJ006I
   ```

4. If a CSECT name does appear, add both the message number and the CSECT name to your keyword string, in one of the formats shown below (in this example, a message number of CSQJ311E and a CSECT name of CSQJC005 are used):

   Free format

   ```
   5655R3600 R700 MSGCSQJ311E CSQJC005
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 MS/CSQJ311E RIDS/CSQJC005
   ```

5. If the message contains return or reason codes, add these to your keyword string, in one of the formats shown below:

   Free format

   ```
   5655R3600 R700 MSGCSQM002I RCE
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 MS/CSQM002I PRCS/0000000E
   ```

6. If the message contains any other types of variables, append them to your keyword string.

   Free format

   ```
   5655R3600 R700 MSGCSQJ104I OPEN
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 MS/CSQJ1041 MS/OPEN
   ```

7. Turn to "Searching the IBM database" on page 75.

Performance keyword

You can resolve most performance problems through system tuning, which should be handled by the WebSphere MQ for z/OS system administrator. Before following the procedure below, use this checklist to verify that the performance problem cannot be resolved through other means:

- Refer to "Dealing with performance problems" on page 27 to see if you can change the way you have designed your WebSphere MQ for z/OS subsystem and applications to improve their performance.
- Verify that the performance problem is not related to a WAIT or LOOP. Turn to "Dealing with waits and loops" on page 22.
- If the problem occurred immediately after you did something a WebSphere MQ manual told you to do, the problem might be related to the manual. Turn to "Documentation keyword" on page 88.

- If performance degraded after someone tuned WebSphere MQ for z/OS, verify that the tuning options selected were appropriate. Perhaps you can resolve the problem by choosing other options.

If you have verified that the performance problem cannot be resolved through other means, use the following procedure:

1. Record the actual performance, expected performance, and source of expected performance criteria.
2. Add PERFM to your keyword string, as shown below, and turn to "Searching the IBM database" on page 75.

   Free format

   ```
   5655R3600 R700 PERFM
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 PERFM
   ```
3. If required, you can narrow your search by adding free-format keywords that describe what you were doing when you experienced the performance problem.

Documentation keyword

The DOC keyword identifies problems caused by incorrect or missing information in a WebSphere MQ manual. It is possible that a documentation problem could be detected when trying to resolve problems with messages, incorrect output, and performance.

Use the following procedure if you need to use the DOC keyword in your keyword string:

1. Locate the incomplete or erroneous information. Note the page, or topic number, and describe the error and the resulting problem.
2. Add the document number, hyphens omitted, to your keyword string, in one of the formats shown below (in this example, the document number for this manual (GC34-6600–00) is used):

   Free format

   ```
   5655R3600 R700 DOC GC34660000
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 PUBS/GC34660000
   ```

   Turn to "Searching the IBM database" on page 75.

   If your search is unsuccessful, follow Step 3.
3. Broaden your search by replacing the last two digits with two asterisks (**). This searches for all problems on that document, rather than on a specific release of the document.

   Free format

   ```
   5655R3600 R700 DOC GC346600**
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 PUBS/GC346600**
   ```

   If your search is unsuccessful, follow Step 4.
4. If the problem is severe, consider initiating a DOC APAR. Use the information gathered in Step 1, and turn to "Resolving a problem" on page 97.

5. If the problem is less severe, include your suggestions on the Readers' Comment Form in the back of that manual, or send it electronically as described at the back of the manual. Include your name and address if you want a reply.

   Corrections resulting from readers' comments are included in future editions of the manual but are not included in the software support database.

Incorrect output keyword

Use the INCORROUT keyword when output was expected but not received, or when output was different from expected. However, if this problem occurred after you did something that a WebSphere MQ manual told you to do, the manual could be in error. If this is the case, refer to "Documentation keyword" on page 88.

1. Add **INCORROUT** to your existing keyword string.

   Free format

   ```
   5655R3600 R700 INCORROUT
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 INCORROUT
   ```

2. Determine the function and secondary modifier keywords for your problem from Table 14 and Table 15.
3. Add the modifier keywords to your string and use it to search the database. See "Searching the IBM database" on page 75.

   Free format

   ```
   5655R3600 R700 INCORROUT RECOVERY BACKOUT
   ```

   Structured format

   ```
   PIDS/5655R3600 LVLS/700 INCORROUT RECOVERY BACKOUT
   ```

Table 14. INCORROUT modifier keywords: RECOVERY

Secondary keywords	Problem occurrence
none	During recovery
BACKOUT	At backout time
CHECKPOINT	At checkpoint time
COMMIT	At commit time
LOGGING	During logging
RECOVER	During attempt to recover in-doubt
RESTART	During restart process

Table 15. INCORROUT modifier keywords: UTILITY

Secondary keywords	Problem occurrence
none	While running a utility
CSQ1LOGP	While running CSQ1LOGP
CHANGE LOG	While using the Change Log Inventory utility or CSQJUFMT
PRINT LOGMAP	While using the Print Log Map utility
COMMAND	While running the COMMAND function or SDEFS
COPY	While running the COPY function or SCOPY
EMPTY	While running the EMPTY function

Table 15. INCORROUT modifier keywords: UTILITY (continued)

Secondary keywords	Problem occurrence
FORMAT	While running the FORMAT function, COPYPAGE, RESETPAGE or PAGEINFO
LOAD	While running the LOAD function
CSQUDLQH	While running the dead.letter.queue handler
CSQ5PQSG	While running CSQ5PQSG

Chapter 5. Working with IBM to solve your problem

Reporting a problem to the IBM software support group

The IBM Product Support Services (PSS) Software Support structure exists to help you resolve problems with IBM products, and to ensure that you can make the best use of your IBM computing systems. Software support is available to all licensed users of IBM licensed programs; you can get assistance by contacting your local support center.

This chapter helps you to decide when to contact the support center, and what information you need to have collected before doing so.

When to contact the support center

Before contacting the support center, try to ensure that the problem belongs with the center. Don't worry if you can't be sure that the problem is due to WebSphere MQ for z/OS itself. How sure you are depends on the complexity of your installation, the experience and skill levels of your systems staff, and the symptoms that your system has been showing.

In practice, quite a lot of errors reported to Software Support turn out to be user errors, or they cannot be reproduced, or they need to be dealt with by other parts of IBM Service such as Hardware Customer Engineering. User errors are mainly caused by bugs in application programs, and mistakes in setting up systems.

Dealing with the support center

If you choose to contact the support center by telephone, your first contact at the support center is the call receipt operator, who takes initial details and puts your problem on a queue. You are subsequently contacted by a support center representative, and your problem is taken from there.

Alternatively, you might have access to an electronic system for reporting problems to the support center. In this case, a support center representative will respond to your communication.

The support center needs to know as much as possible about your problem, so have the information ready before contacting them. If contacting them by telephone, it is a good idea to write the information on a problem reporting sheet such as the one shown in Figure 17 on page 92.

There are two advantages of using a problem reporting sheet when contacting the IBM support center:

- In a telephone conversation, you will be better prepared to respond to the questions that you might be asked if you have all your findings before you on a sheet of paper.
- You can use the information for planning, organizing, and establishing priorities for controlling and resolving these problems.

```
PROBLEM REPORTING SHEET

Date                    Severity                Problem  No.

                                                Incident No.

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Problem/Enquiry

Abend                   Incorrout               z/OS Release

Wait                    Module                  WebSphere MQ Release

Loop                    Message                 CICS Release

Performance             Other                   IMS Release

                                                DB2 Release
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Documentation available

Abend                   System dump             Program output

Message                 Transaction dump        Other

Trace                   Translator output

Symptom string          Compiler output

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Actions

Date          Name              Activity









- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Resolution

APAR                    PTF                     Other
```

Figure 17. Sample problem reporting sheet

If you use an electronic system for reporting problems to the support center, you should still include as much information as possible about your problem.

You should also maintain your own in-house tracking system for problems. A problem tracking system records and documents all problems.

What the support center needs to know

When you contact the support center, whether by telephone or electronically, you need to state the name of your organization and your *access code*. Your access code

is a unique code authorizing you to use IBM PSS Software Support, and you provide it every time you contact the center. This information is used to access your customer profile, which contains information about your address, relevant contact names, telephone numbers, and details of the IBM products at your installation.

The support center needs to know whether this is a new problem, or a further communication regarding an existing one. If it is new, it is assigned a unique *incident number*. A *problem management record* (PMR) is opened on the RETAIN system, where all activity associated with your problem is recorded. The problem remains 'open' until you are in agreement with the support center that it has been resolved and can now be closed.

Make a note of the incident number on your own problem reporting sheet. The support center expects you to quote the incident number in all future communications connected with this problem.

If the problem is new to you, you need to state the source of the problem within your system software—that is, the program that seems to be the cause of the problem. Because you are reading this manual, it is likely that you have already identified WebSphere MQ for z/OS as the problem source. You also have to give the version and release number.

You need to give the *severity level* for the problem. Severity levels can be 1, 2, 3, or 4 and they have the following meanings:

Level 1
: This indicates that you cannot use the system, and have a critical condition that needs immediate attention.

Level 2
: This indicates that you can use the system, but that operation is severely restricted.

Level 3
: This indicates that you can use the program, with limited functions, but the problem is not critical to your overall operation.

Level 4
: This indicates that you have found a way to work around the problem; however, further action is required to correct the problem.

When deciding the severity of the problem, take care neither to understate it, nor to overstate it. The support center procedures depend on the severity level so that the most appropriate use can be made of the center's skills and resources. A severity level 1 problem is normally dealt with immediately.

Next, you need to state a brief description of the problem. You might also be asked to quote the WebSphere MQ for z/OS symptom string, or to give any keywords associated with the problem. The primary keywords are ABEND, WAIT, LOOP, PERFM, INCORROUT, MSG, and DOC, corresponding exactly with the problem classification types used in "Building a keyword string" on page 78. Strings containing other keywords are also useful. These are not predefined, and might include such items as a message or message number, an abend code, any parameters known to be associated with the problem, or, for example, STARTUP or INITIALIZATION.

The keywords are subsequently used as search arguments on the RETAIN database, to see if your problem is a known one that has already been the subject of an *authorized program analysis report* (APAR).

Finally, let the support center know if any of the following events occurred before the problem appeared:

- Changes in the level of z/OS or licensed programs
- Regenerations
- PTFs applied
- Additional features used
- Application programs changed
- Unusual operator action

You might be asked to give values from a formatted dump or trace table, or to carry out some special activity, for example to set a trap, or to use trace with a specific type of selectivity, and then to report the results.

Note: You will be given guidance by the support center on how to obtain this information.

How your problem is subsequently progressed depends on its nature. The representative who handles the problem gives you guidance on what is required from you. The possibilities are described in the next section.

What happens next

Details of your problem are passed to the appropriate support group using the RETAIN problem management system. Your problem, assuming it is one associated with WebSphere MQ for z/OS, is put on the WebSphere MQ for z/OS queue. The problems are dealt with in order of receipt and severity level.

At first, an IBM support center representative uses the keywords that you have provided to search the RETAIN database. If your problem is found to be one already known to IBM, and a fix has been devised for it, a *program temporary fix* (PTF) can be dispatched to you quickly. Alternatively, you might be asked to try running your installation using different settings.

If the RETAIN search is unsuccessful, you are asked to provide more information about your problem. Guidance on collecting and sending documentation to IBM is given in "Collecting documentation for the problem" on page 95 and "Sending the documentation to the change team" on page 96.

If the problem requires a change to the WebSphere MQ for z/OS code or documentation, an *authorized program analysis report* (APAR) is submitted. This is dealt with by the WebSphere MQ for z/OS Development Support Group or *change team* and provides a means of tracking the change.

It might be necessary to have several follow-up communications, depending on the complexity of the symptoms and your system environment. In every case, the actions taken by you and the support center are entered in the original PMR. The representative can then be acquainted with the full history of the problem before the next communication.

Collecting documentation for the problem

As a general rule, the documentation you need to submit for a problem includes all the material you need yourself to do problem determination. Some of the documentation is common to all WebSphere MQ for z/OS problems, and some is specific to particular types of problem.

Make sure the problem you have described can be seen in the documentation you send. If the problem has ambiguous symptoms, you need to reveal the sequence of events leading to the failure. Tracing is valuable in this respect but you need to provide details that trace cannot give. You are encouraged to annotate your documentation, if your annotation is legible and does not cover up vital information. You can highlight any data in hardcopy you send, using transparent highlighting markers. You can also write notes in the margins, preferably using a red pen so that the notes are not overlooked.

Finally, note that if you send too little documentation, or if it is unreadable, the change team will have to return your problem marked "insufficient documentation". It is, therefore, worthwhile preparing your documentation carefully and sending everything relevant to the problem.

The general documentation is described below. However, these are only guidelines, you must find out from the IBM support center representative precisely what documentation you need to send for your specific problem.

General documentation needed for all problems with WebSphere MQ for z/OS

Here is a list of the general documentation you might be asked to submit for a PMR:

- Any hardcopy or softcopy illustrating the symptoms of the problem.
- The dump of the problem, see "Getting a dump" on page 48.
- The appropriate SYS1.LOGREC records, see "SYS1.LOGREC information" on page 64.
- The system log.
- A portion of the WebSphere MQ for z/OS job log.
- Trace records, see "Using trace for problem determination" on page 65.
- Trace information produced by the CICS or IMS adapter, see "The CICS adapter trace" on page 74.
- Buffer pool statistics, see the sections on using SMF and type 115 records in the WebSphere MQ for z/OS System Setup Guide.
- Listings of relevant application programs.
- A list of PTFs and APARs applied.

  Use the sample JCL below to produce a list of all the PTFs, APARs, user modifications and product code that has been installed in the global zone specified by SMPCSI:

```
//SMPELIST  EXEC PGM=GIMSMP,REGION=4096K
//SMPCSI    DD  DSN=shlqual.global.csi,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SMPCNTL   DD *
    SET BOUNDARY(GLOBAL) .
    LIST
      SYSMODS
         APARS
```

```
        FUNCTIONS
        PTFS
        USERMODS
    ALLZONES .
/*
```

- Dump of Coupling Facility administration structure, see Figure 6 on page 51.
- Dump of Coupling Facility application structure, see Figure 6 on page 51.
- Dump of other queue managers in queue-sharing group, see Figure 2 on page 50.
- Definitions of the objects in your system.

  You can find these by using the DISPLAY command for each type of object, for example:

  ```
  DISPLAY QUEUE (*) ALL
  ```

  Or, if you regularly make a backup of your object using the MAKEDEF feature of CSQUTIL COMMAND function, the output from that backup job.
- Your WebSphere MQ DB2 tables.

  You can get these by using the sample job CSQ45STB in thlqual.SCSQPROCS to produce a report of all the DB2 tables used by WebSphere MQ.

Because of the size of SVC dumps in the cross memory environment, transfer the SYS1.DUMPxx data set to a tape or like device. You can use the PRDMP service aid program to transfer the SYS1.DUMPxx data set contents to another data set for archiving until the problem is resolved. Alternatively, your support center representative might give you the address of an FTP site where you can send your dump electronically.

Depending on the nature of the problem, the IBM support center might ask you to send the entire dump on tape. This allows the support center to extract any additional data needed to resolve the problem; for example, CSA, SQA, or the private storage area.

Sending the documentation to the change team

When submitting documentation for your problem, your IBM support center will advise you on the most appropriate method to use. Please contact them for details.

Each item submitted must have the following information attached and visible:

- The PMR number assigned by IBM
- A list of data sets on the tape (application source program, JCL, or data)
- A list of how the tape was made, including:
  - The exact JCL listing or the commands used
  - The recording mode and density
  - Tape labeling
  - The record format, logical record length, and block size used for each data set

When the change team receives the package, this is noted on your PMR record on the RETAIN system. The team then investigates the problem. Sometimes, they will ask for more documentation, perhaps specifying some trap you must apply to get it.

When you are satisfied that the problem is solved, a code is entered on RETAIN to close the PMR, and if necessary, you are provided with a fix.

You can enquire any time at your support center on how your PMR is progressing, particularly if it is a problem of high severity.

Resolving a problem

An *authorized program analysis report* (APAR) is the means by which a problem with an IBM program is documented, tracked, and corrected. It is also used to track problems with IBM documents.

An APAR is raised by the IBM change team when a new problem is reported for which a program or documentation change is required. It is separate to the PMR that is raised when you report first report the problem.

When the change team solves the problem, they might produce a local fix enabling you to get your system running properly again. Finally, a *program temporary fix* (PTF) is produced to replace the module in error, and the APAR is closed.

The APAR process

The first step in the APAR process is that an IBM support center representative enters your APAR into the RETAIN system. The APAR text contains a description of your problem. If you have found a means of getting round the problem, details of this are entered as well. Your name is also entered, so that the support center know whom to contact if the change team needs to ask anything further about the APAR documentation.

When the APAR has been entered, you are given an APAR number. You must write this number on all the documentation you submit to the change team. This number is always associated with the APAR and its resolution and, if a code change is required, with the fix as well.

During the APAR process, the change team might ask you to test the fix on your system.

Lastly, you need to apply the PTF resulting from the APAR when it becomes available.

Applying the fix

When the change team have created a fix for your problem, they might want you to test it on your system.

When the team is confident that the fix is satisfactory, the APAR is certified and the APAR is closed.

Occasionally, the solution to the APAR requires a change to the documentation only. In some circumstances, the APAR might be closed with a classification code of **FIN**, which means that if there is a subsequent release of WebSphere MQ for z/OS, a fix for this problem will be provided at this time.

The APAR becomes a PTF

If the solution requires a code change to the current release, when the APAR is closed the change is distributed as a PTF.

If you want a PTF to resolve a specific problem, you can order it explicitly by its PTF number through the IBM support center. Otherwise, you can wait for the PTF to be sent out on the standard distribution tape.

Chapter 6. SDB format symptom-to-keyword cross reference

Structured database (SDB) format is one of the formats that you can use for searching the RETAIN database. Structured symptoms are also called RETAIN symptoms and "failure keywords". Table 16 lists which prefix keyword to use for which symptom.

"Searching the IBM database" on page 75 provides details about searching RETAIN.

Table 16. SDB format symptom-to-keyword cross-reference

Symptom	Keyword	Symptom	Keyword
abend	AB/	access method	RIDS/
address	ADRS/	APAR	PTFS/
assembler macro	RIDS/	assembler message	MS/
CLIST	RIDS/	command	PCSS/
compiler message	MS/	completion code	PRCS/
component	PIDS/	condition code	PRCS/
control block	FLDS/	control block offset	ADRS/
control register	REGS/	CSECT	RIDS/
data set name	PCSS/	dependent component	PIDS/
device error code	PRCS/	disabled wait (coded)	WS/
displacement	ADRS/	display	DEVS/
document	PUBS/	DSECT	FLDS/
enabled wait (coded)	WS/	error code	PRCS/
EXEC	RIDS/	feedback code	PRCS/
field	FLDS/	field value	VALU/
file mode	PCSS/	file name	PCSS/
file type	PCSS/	flag	FLDS/
floating-point register	REGS/	full-screen mode	PCSS/
function key	PCSS/	general purpose register	REGS/
hang	WS/	hung user or task	WS/
I/O operator codes	OPCS/	incorrect output	INCORROUT*
JCL card	PCSS/	JCL parameter	PCSS/
job step code	PRCS/	key	PCSS/
label, code	FLDS/	language statement	PCSS/
level	LVLS/	library name	PCSS/
line command	PCSS/	loop	LOOP*
low core address	ADRS/	machine check	SIG/
macro as a routine	RIDS/	macro as a statement	PCSS/
maintenance level	PTFS/	message	MS/
module	RIDS/	offset	ADRS/

Table 16. SDB format symptom-to-keyword cross-reference (continued)

Symptom	Keyword	Symptom	Keyword
opcode	OPCS/	operator command	PCSS/
operator key	PCSS/	operator message	MS/
option	PCSS/	overlay	OVS/
PA key	PCSS/	panel	RIDS/
parameter	PCSS/	performance	PERFM*
PF key	PCSS/	procedure name	PCSS/
process name	PCSS/	profile option	PCSS/
program check	AB/	program id	RIDS/
program key	PCSS/	program statement	PCSS/
PSW	FLDS/	PTF, PE or otherwise	PTFS/
publication	PUBS/	PUT level	PTFS/
reason code	PRCS/	register value	VALU/
register	REGS/	release level	LVLS/
reply to message	PCSS/	reply to prompt	PCSS/
request code	OPCS/	response to message	PCSS/
response to prompt	PCSS/	return code	PRCS/
routine	RIDS/	service level	PTFS/
special character	PCSS/	SRL	PUBS/
statement	PCSS/	status code	PRCS/
step code	PRCS/	structure word	FLDS/
subroutines	RIDS/	SVC	OPCS/
SYSGEN parameter	PCSS/	system check	PRCS/
table	FLDS/	terminal key	PCSS/
value	VALU/	variable	FLDS/
wait (coded)	WS/	wait (uncoded)	WAIT*
Note: An asterisk (*) indicates that there is no prefix keyword for this type of problem. Use the type-of-failure keyword shown for searches of the software support database.			

Chapter 7. WebSphere MQ component and resource manager identifiers

The component-identifier keyword identifies the library within the IBM software support database that contains *authorized program analysis reports* (APARs) and *program temporary fixes* (PTFs) for the product. Resource manager identifiers (RMIDs) are used to limit the volume of data collected in a trace.

Table 17. WebSphere MQ component and resource manager identifiers

ID	Prefix	Hex ID	Component name	RMID
—	AMT	—	AMI	—
—	CMQ	—	Application header files	—
m	CSQm	X'94'	Connection manager	148
t	CSQt	X'A3'	Topic manager	163
A	CSQA	X'C1'	Application interface	—
B	CSQB	X'C2'	Batch adapter	—
C	CSQC	X'C3'	CICS adapter	—
E	CSQE	X'C5'	Coupling Facility manager	197
F	CSQF	X'C6'	Message generator	24
G	CSQG	X'C7'	Functional recovery manager	199
H	CSQH	X'C8'	Security manager interface	200
I	CSQI	X'C9'	Data manager	201
J	CSQJ	X'D1'	Recovery log manager	4
L	CSQL	X'D3'	Lock manager	211
M	CSQM	X'D4'	Message manager	212
N	CSQN	X'D5'	Command server	213
O	CSQO	X'D6'	Operations and control	—
P	CSQP	X'D7'	Buffer manager	215
Q	CSQQ	X'D8'	IMS adapter	—
R	CSQR	X'D9'	Recovery manager	3
S	CSQS	X'E2'	Storage manager	6
T	CSQT	X'E3'	Timer services	227
U	CSQU	X'E4'	Utilities	—
V	CSQV	X'E5'	Agent services	2
W	CSQW	X'E6'	Instrumentation facilities	16, 26
X	CSQX	X'E7'	Distributed queuing	231
Y	CSQY	X'E8'	Initialization procedures and general services	1
Z	CSQZ	X'E9'	System parameter manager	12
1	CSQ1	X'F1'	Service facilities	—
2	CSQ2	X'F2'	WebSphere MQ-IMS bridge	242
3	CSQ3	X'F3'	Subsystem support	7, 8
4	CSQ4	X'F4'	Sample programs	—

Table 17. WebSphere MQ component and resource manager identifiers (continued)

ID	Prefix	Hex ID	Component name	RMID
5	CSQ5	X'F5'	DB2 manager	245
6	CSQ6	X'F6'	Customization	—
7	CSQ7	X'F7'	Dump formatting	—
8	CSQ8	X'F8'	Installation	—
9	CSQ9	X'F9'	Generalized command preprocessor	23
—	IMQ	—	C++ bindings	—

Chapter 8. CICS adapter trace entries

The CICS trace entry for these values is AP0xxx (where xxx is the hexadecimal equivalent of the trace number you specified when the CICS adapter was enabled). These trace entries are all issued by CSQCTRUE, except CSQCTEST, which is issued by CSQCRST and CSQCDSP.

Table 18. CICS adapter trace entries

Name	Description	Trace sequence	Trace data
CSQCABNT	Abnormal termination	Before issuing END_THREAD ABNORMAL to WebSphere MQ. This is due to the end of the task and therefore an implicit backout could be performed by the application. A ROLLBACK request is included in the END_THREAD call in this case.	Unit of work information. You can use this information when finding out about the status of work. (For example, it can be verified against the output produced by the DISPLAY THREAD command, or the log print utility.)
CSQCAUID	Bridge security	Before validating bridge user password or PassTicket.	User ID.
CSQCBACK	Syncpoint backout	Before issuing BACKOUT to WebSphere MQ. This is due to an explicit backout request from the application.	Unit of work information.
CSQCCONX	MQCONNX	Before issuing **MQCONNX** to WebSphere MQ.	Connection tag.
CSQCCCRC	Completion code and reason code	After unsuccessful return from API call.	Completion code and reason code.
CSQCCOMM	Syncpoint commit	Before issuing COMMIT to WebSphere MQ. This can be due to a single-phase commit request or the second phase of a two-phase commit request. The request is due to a explicit syncpoint request from the application.	Unit of work information.
CSQCDCFF	IBM use only		
CSQCDCIN	IBM use only		
CSQCDCOT	IBM use only		
CSQCEXER	Execute resolve	Before issuing EXECUTE_RESOLVE to WebSphere MQ.	The unit of work information of the unit of work issuing the EXECUTE_RESOLVE. This is the last in-doubt unit of work in the resynchronization process.
CSQCGETW	GET wait	Before issuing CICS wait.	Address of the ECB to be waited on.
CSQCGMGD	GET message data	After successful return from **MQGET**.	Up to 40 bytes of the message data.
CSQCGMGH	GET message handle	Before issuing **MQGET** to WebSphere MQ.	Object handle.
CSQCGMGI	Get message ID	After successful return from **MQGET**.	Message ID and correlation ID of the message.
CSQCHCER	Hconn error	Before issuing any MQ verb.	Connection handle.

Table 18. CICS adapter trace entries (continued)

Name	Description	Trace sequence	Trace data
CSQCINDL	In-doubt list	After successful return from the second INQUIRE_INDOUBT.	The in-doubt units of work list.
CSQCINDO	IBM use only		
CSQCINDS	In-doubt list size	After successful return from the first INQUIRE_INDOUBT and the in-doubt list is not empty.	Length of the list; divided by 64 gives the number of in-doubt units of work.
CSQCINDW	Syncpoint in doubt	During syncpoint processing, CICS is in doubt as to the disposition of the unit of work.	Unit of work information.
CSQCINQH	INQ handle	Before issuing **MQINQ** to WebSphere MQ.	Object handle.
CSQCLOSH	CLOSE handle	Before issuing **MQCLOSE** to WebSphere MQ.	Object handle.
CSQCLOST	Disposition lost	During the resynchronization process, CICS informs the adapter that it has been cold started so no disposition information regarding the unit of work being resynchronized is available.	Unit of work ID known to CICS for the unit of work being resynchronized.
CSQCNIND	Disposition not in doubt	During the resynchronization process, CICS informs the adapter that the unit of work being resynchronized should not have been in doubt (that is, perhaps it is still running).	Unit of work ID known to CICS for the unit of work being resynchronized.
CSQCNORT	Normal termination	Before issuing END_THREAD NORMAL to WebSphere MQ. This is due to the end of the task and therefore an implicit syncpoint commit might be performed by the application. A COMMIT request is included in the END_THREAD call in this case.	Unit of work information.
CSQCOPNH	OPEN handle	After successful return from **MQOPEN**.	Object handle.
CSQCOPNO	OPEN object	Before issuing **MQOPEN** to WebSphere MQ.	Object name.
CSQCPMGD	PUT message data	Before issuing **MQPUT** to WebSphere MQ.	Up to 40 bytes of the message data.
CSQCPMGH	PUT message handle	Before issuing **MQPUT** to WebSphere MQ.	Object handle.
CSQCPMGI	PUT message ID	After successful **MQPUT** from WebSphere MQ.	Message ID and correlation ID of the message.
CSQCPREP	Syncpoint prepare	Before issuing PREPARE to WebSphere MQ in the first phase of two-phase commit processing. This call can also be issued from the distributed queuing component as an API call.	Unit of work information.
CSQCP1MD	PUTONE message data	Before issuing **MQPUT1** to WebSphere MQ.	Up to 40 bytes of data of the message.

Table 18. CICS adapter trace entries (continued)

Name	Description	Trace sequence	Trace data
CSQCP1MI	PUTONE message ID	After successful return from **MQPUT1**.	Message ID and correlation ID of the message.
CSQCP1ON	PUTONE object name	Before issuing **MQPUT1** to WebSphere MQ.	Object name.
CSQCRBAK	Resolved backout	Before issuing RESOLVE_ROLLBACK to WebSphere MQ.	Unit of work information.
CSQCRCMT	Resolved commit	Before issuing RESOLVE_COMMIT to WebSphere MQ.	Unit of work information.
CSQCRMIR	RMI response	Before returning to the CICS RMI (resource manager interface) from a specific invocation.	Architected RMI response value. Its meaning depends of the type of the invocation. To determine the type of invocation, look at previous trace entries produced by the CICS RMI component.
CSQCRSYN	Resync	Before the resynchronization process starts for the task.	Unit of work ID known to CICS for the unit of work being resynchronized.
CSQCSETH	SET handle	Before issuing **MQSET** to WebSphere MQ.	Object handle.
CSQCTASE	IBM use only		
CSQCTEST	Trace test	Used in EXEC CICS ENTER TRACE call to verify the trace number supplied by the user or the trace status of the connection.	No data.

Chapter 9. Examples of CEDF output

This appendix gives examples of the output produced by the CICS execution diagnostic facility (CEDF) when using WebSphere MQ. The examples show the data produced on entry to and exit from the following MQI calls, in both hexadecimal and character format:

- "MQOPEN"
- "MQCLOSE" on page 108
- "MQPUT" on page 109
- "MQPUT1" on page 111
- "MQGET" on page 112
- "MQINQ" on page 113
- "MQSET" on page 115

Other MQI calls produce similar data.

MQOPEN

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object descriptor
ARG 002	Options
ARG 003	Object handle
ARG 004	Completion code
ARG 005	Reason code

```
  STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000000200004044')            AT X'05ECAFD8'
 001: ARG 001 (X'D6C4404000000010000000 1C3C5C4C6')            AT X'00144910'
 001: ARG 002 (X'00000072000000000000000000000000')            AT X'001445E8'
 001: ARG 003 (X'00000000000000720000000000000000')            AT X'001445E4'
 001: ARG 004 (X'00000000000000000000000000000000')            AT X'001445EC'
 001: ARG 005 (X'00000000000000000000000000000000')            AT X'001445F0'
```

Figure 18. Example CEDF output on entry to an MQOPEN call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000000200004044')            AT X'05ECAFD8'
 001: ARG 001 (X'D6C4404000000010000000 1C3C5C4C6')            AT X'00144910'
 001: ARG 002 (X'00000072000000000000000000000000')            AT X'001445E8'
 001: ARG 003 (X'00000001000000720000000000000000')            AT X'001445E4'
 001: ARG 004 (X'00000000000000000000000000000000')            AT X'001445EC'
 001: ARG 005 (X'00000000000000000000000000000000')            AT X'001445F0'
```

Figure 19. Example CEDF output on exit from an MQOPEN call (hexadecimal)

```
 STATUS:  ABOUT TO EXECUTE COMMAND
 CALL TO RESOURCE MANAGER MQM
  001: ARG 000 ('.............. .')
  001: ARG 001 ('OD  ........CEDF')
  001: ARG 002 ('................')
  001: ARG 003 ('................')
  001: ARG 004 ('................')
  001: ARG 005 ('................')
```

Figure 20. Example CEDF output on entry to an MQOPEN call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('.............. .')
 001: ARG 001 ('OD  ........CEDF')
 001: ARG 002 ('................')
 001: ARG 003 ('................')
 001: ARG 004 ('................')
 001: ARG 005 ('................')
```

Figure 21. Example CEDF output on exit from an MQOPEN call (character)

MQCLOSE

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object handle
ARG 002	Options
ARG 003	Completion code
ARG 004	Reason code

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'00000001000000720000000000000000')            AT X'001445E4'
 001: ARG 002 (X'00000000000000010000000200004044')            AT X'05ECAFD8'
 001: ARG 003 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 004 (X'00000000000000080000000800000060')            AT X'001445F0'
```

Figure 22. Example CEDF output on entry to an MQCLOSE call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000000000007200000000')            AT X'001445E0'
 001: ARG 001 (X'00000000000000720000000000000000')            AT X'001445E4'
 001: ARG 002 (X'00000000000000010000000200004044')            AT X'05ECAFD8'
 001: ARG 003 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 004 (X'00000000000000080000000800000060')            AT X'001445F0'
```

Figure 23. Example CEDF output on exit from an MQCLOSE call (hexadecimal)

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('................')
 001: ARG 002 ('.............. .')
 001: ARG 003 ('................')
 001: ARG 004 ('...............-')
```

Figure 24. Example CEDF output on entry to an MQCLOSE call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('................')
 001: ARG 002 ('.............. .')
 001: ARG 003 ('................')
 001: ARG 004 ('...............-')
```

Figure 25. Example CEDF output on exit from an MQCLOSE call (character)

MQPUT

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object handle
ARG 002	Message descriptor
ARG 003	Put message options
ARG 004	Buffer length
ARG 005	Message data
ARG 006	Completion code
ARG 007	Reason code

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'00000001000000720000000000000000')            AT X'001445E4'
 001: ARG 002 (X'D4C4404000000001000000000000000800')          AT X'001449B8'
 001: ARG 003 (X'D7D4D64000000001000000240000000')             AT X'00144B48'
 001: ARG 004 (X'00000008000000000000000000040000')            AT X'001445F4'
 001: ARG 005 (X'5C5CC8C5D3D3D640E6D6D9D3C45C5C5C')            AT X'00144BF8'
 001: ARG 006 (X'00000000000000000000000800000000')            AT X'001445EC'
 001: ARG 007 (X'00000000000000080000000000000000')            AT X'001445F0'
```

Figure 26. Example CEDF output on entry to an MQPUT call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'00000001000000720000000000000000')            AT X'001445E4'
 001: ARG 002 (X'D4C4404000000001000000000000000800')          AT X'001449B8'
 001: ARG 003 (X'D7D4D64000000001000000240000000')             AT X'00144B48'
 001: ARG 004 (X'00000008000000000000000000040000')            AT X'001445F4'
 001: ARG 005 (X'5C5CC8C5D3D3D640E6D6D9D3C45C5C5C')            AT X'00144BF8'
 001: ARG 006 (X'00000000000000000000000800000000')            AT X'001445EC'
 001: ARG 007 (X'00000000000000080000000000000000')            AT X'001445F0'
```

Figure 27. Example CEDF output on exit from an MQPUT call (hexadecimal)

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('................')
 001: ARG 002 ('MD  ............')
 001: ARG 003 ('PMO ............')
 001: ARG 004 ('................')
 001: ARG 005 ('**HELLO WORLD***')
 001: ARG 006 ('................')
 001: ARG 007 ('................')
```

Figure 28. Example CEDF output on entry to an MQPUT call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('................')
 001: ARG 002 ('MD  ............')
 001: ARG 003 ('PMO ............')
 001: ARG 004 ('................')
 001: ARG 005 ('**HELLO WORLD***')
 001: ARG 006 ('................')
 001: ARG 007 ('................')
```

Figure 29. Example CEDF output on exit from an MQPUT call (character)

MQPUT1

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object descriptor
ARG 002	Message descriptor
ARG 003	Put message options
ARG 004	Buffer length
ARG 005	Message data
ARG 006	Completion code
ARG 007	Reason code

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000000000007200000000')            AT X'001445E0'
 001: ARG 001 (X'D6C4404000000010000001C3C5C4C6')              AT X'00144910'
 001: ARG 002 (X'D4C4404000000010000000000000008')             AT X'001449B8'
 001: ARG 003 (X'D7D4D64000000010000002400000000')             AT X'00144B48'
 001: ARG 004 (X'00000008000000080000006000040000')            AT X'001445F4'
 001: ARG 005 (X'5C5CC8C5D3D3D640E6D6D9D3C45C5C5C')            AT X'00144BF8'
 001: ARG 006 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 007 (X'00000000000000080000000800000060')            AT X'001445F0'
```

Figure 30. Example CEDF output on entry to an MQPUT1 call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000000000007200000000')            AT X'001445E0'
 001: ARG 001 (X'D6C4404000000010000001C3C5C4C6')              AT X'00144910'
 001: ARG 002 (X'D4C4404000000010000000000000008')             AT X'001449B8'
 001: ARG 003 (X'D7D4D64000000010000002400000000')             AT X'00144B48'
 001: ARG 004 (X'00000008000000080000006000040000')            AT X'001445F4'
 001: ARG 005 (X'5C5CC8C5D3D3D640E6D6D9D3C45C5C5C')            AT X'00144BF8'
 001: ARG 006 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 007 (X'00000000000000080000000800000060')            AT X'001445F0'
```

Figure 31. Example CEDF output on exit from an MQPUT1 call (hexadecimal)

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('OD  ........CEDF')
 001: ARG 002 ('MD  ............')
 001: ARG 003 ('PMO ............')
 001: ARG 004 ('...........-....')
 001: ARG 005 ('**HELLO WORLD***')
 001: ARG 006 ('................')
 001: ARG 007 ('...............-')
```

Figure 32. Example CEDF output on entry to an MQPUT1 call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('OD  ........CEDF')
 001: ARG 002 ('MD  ............')
 001: ARG 003 ('PMO ............')
 001: ARG 004 ('...........-....')
 001: ARG 005 ('**HELLO WORLD***')
 001: ARG 006 ('................')
 001: ARG 007 ('...............-')
```

Figure 33. Example CEDF output on exit from an MQPUT1 call (character)

MQGET

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object handle
ARG 002	Message descriptor
ARG 003	Get message options
ARG 004	Buffer length
ARG 005	Message buffer
ARG 006	Message length
ARG 007	Completion code
ARG 008	Reason code

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'0000000100000072000000000000000000')          AT X'001445E4'
 001: ARG 002 (X'D4C4404000000001000000000000000000')          AT X'001449B8'
 001: ARG 003 (X'C7D4D6400000000100004044FFFFFFFF')            AT X'00144B00'
 001: ARG 004 (X'00000008000000000000000000040000')            AT X'001445F4'
 001: ARG 005 (X'00000000000000000000000000000000')            AT X'00144C00'
 001: ARG 006 (X'00000000000000000004000000000000')            AT X'001445F8'
 001: ARG 007 (X'00000000000000000000000800000000')            AT X'001445EC'
 001: ARG 008 (X'00000000000000080000000000000000')            AT X'001445F0'
```

Figure 34. Example CEDF output on entry to an MQGET call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'0000000100000072000000000000000000')          AT X'001445E4'
 001: ARG 002 (X'D4C4404000000001000000000000000008')          AT X'001449B8'
 001: ARG 003 (X'C7D4D64000000001000004044FFFFFFFF')           AT X'00144B00'
 001: ARG 004 (X'00000008000000080000000000040000')            AT X'001445F4'
 001: ARG 005 (X'5C5CC8C5D3D3D640E6D6D9D3C45C5C5C')            AT X'00144C00'
 001: ARG 006 (X'00000008000000000000400000000000')            AT X'001445F8'
 001: ARG 007 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 008 (X'00000000000000080000000800000000')            AT X'001445F0'
```

Figure 35. Example CEDF output on exit from an MQGET call (hexadecimal)

```
 STATUS:  ABOUT TO EXECUTE COMMAND
 CALL TO RESOURCE MANAGER MQM
  001: ARG 000 ('................')
  001: ARG 001 ('................')
  001: ARG 002 ('MD  ............')
  001: ARG 003 ('GMO ...... .....')
  001: ARG 004 ('................')
  001: ARG 005 ('................')
  001: ARG 006 ('................')
  001: ARG 007 ('................')
  001: ARG 008 ('................')
```

Figure 36. Example CEDF output on entry to an MQGET call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('................')
 001: ARG 002 ('MD  ............')
 001: ARG 003 ('GMO ...... .....')
 001: ARG 004 ('................')
 001: ARG 005 ('**HELLO WORLD***')
 001: ARG 006 ('................')
 001: ARG 007 ('................')
 001: ARG 008 ('................')
```

Figure 37. Example CEDF output on exit from an MQGET call (character)

MQINQ

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object handle
ARG 002	Count of selectors
ARG 003	Array of attribute selectors
ARG 004	Count of integer attributes
ARG 005	Integer attributes
ARG 006	Length of character attributes buffer
ARG 007	Character attributes

ARG 008	Completion code
ARG 009	Reason code

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000000200004044')            AT X'05ECAFCC'
 001: ARG 001 (X'0000000100000072000000000000000'0')           AT X'001445E4'
 001: ARG 002 (X'00000020000404485ECA00885ECA220')             AT X'05ECAFD4'
 001: ARG 003 (X'0000000D0000000C0000000000000000')            AT X'00144C08'
 001: ARG 004 (X'00000020000404485ECA00885ECA220')             AT X'05ECAFD4'
 001: ARG 005 (X'00000000000000000000000000000000')            AT X'00144C10'
 001: ARG 006 (X'00000000000000010000000200004044')            AT X'05ECAFCC'
 001: ARG 007 (X'00000000000000000000000000000000')            AT X'00144C18'
 001: ARG 008 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 009 (X'00000000000000080000000800040000')            AT X'001445F0'
```

Figure 38. Example CEDF output on entry to an MQINQ call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000000200004044')            AT X'05ECAFCC'
 001: ARG 001 (X'0000000100000072000000000000000'0')           AT X'001445E4'
 001: ARG 002 (X'00000020000404485ECA00885ECA220')             AT X'05ECAFD4'
 001: ARG 003 (X'0000000D0000000C0040000000000000')            AT X'00144C08'
 001: ARG 004 (X'00000020000404485ECA00885ECA220')             AT X'05ECAFD4'
 001: ARG 005 (X'00400000000000000000000000000000')            AT X'00144C10'
 001: ARG 006 (X'00000000000000010000000200004044')            AT X'05ECAFCC'
 001: ARG 007 (X'00000000000000000000000000000000')            AT X'00144C18'
 001: ARG 008 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 009 (X'00000000000000080000000800040000')            AT X'001445F0'
```

Figure 39. Example CEDF output on exit from an MQINQ call (hexadecimal)

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('.............. .')
 001: ARG 001 ('................')
 001: ARG 002 ('...... .e...e.s.')
 001: ARG 003 ('................')
 001: ARG 004 ('...... .e...e.s.')
 001: ARG 005 ('................')
 001: ARG 006 ('.............. .')
 001: ARG 007 ('................')
 001: ARG 008 ('................')
 001: ARG 009 ('................')
```

Figure 40. Example CEDF output on entry to an MQINQ call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('.............. .')
 001: ARG 001 ('................')
 001: ARG 002 ('...... .e...e.s.')
 001: ARG 003 ('......... ......')
 001: ARG 004 ('...... .e...e.s.')
 001: ARG 005 ('. ..............')
 001: ARG 006 ('.............. .')
 001: ARG 007 ('................')
 001: ARG 008 ('................')
 001: ARG 009 ('................')
```

Figure 41. Example CEDF output on exit from an MQINQ call (character)

MQSET

The parameters for this call are:

ARG 000	Connection handle
ARG 001	Object handle
ARG 002	Count of selectors
ARG 003	Array of attribute selectors
ARG 004	Count of integer attributes
ARG 005	Integer attributes
ARG 006	Length of character attributes buffer
ARG 007	Character attributes
ARG 008	Completion code
ARG 009	Reason code

```
STATUS:  ABOUT TO EXECUTE COMMAND
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'0000000100000072000000000000000')             AT X'001445E4'
 001: ARG 002 (X'0000000100000002000040485ECA008')             AT X'05ECAFDC'
 001: ARG 003 (X'00000018000007DF0000000000000000')            AT X'00144C08'
 001: ARG 004 (X'0000000100000002000040485ECA008')             AT X'05ECAFDC'
 001: ARG 005 (X'00000000000000000000000000000000')            AT X'00144C10'
 001: ARG 006 (X'00000000000000010000000200004044')            AT X'05ECAFD8'
 001: ARG 007 (X'00000000000000000000000000000000')            AT X'00144C18'
 001: ARG 008 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 009 (X'00000000000000080000000800000060')            AT X'001445F0'
```

Figure 42. Example CEDF output on entry to an MQSET call (hexadecimal)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 (X'00000000000000010000007200000000')            AT X'001445E0'
 001: ARG 001 (X'00000001000000720000000000000000')            AT X'001445E4'
 001: ARG 002 (X'00000001000000020000404485ECA008')            AT X'05ECAFDC'
 001: ARG 003 (X'00000018000007DF0000000000000000')            AT X'00144C08'
 001: ARG 004 (X'00000001000000020000404485ECA008')            AT X'05ECAFDC'
 001: ARG 005 (X'00000000000000000000000000000000')            AT X'00144C10'
 001: ARG 006 (X'00000000000000010000000200004044')            AT X'05ECAFD8'
 001: ARG 007 (X'00000000000000000000000000000000')            AT X'00144C18'
 001: ARG 008 (X'00000000000000000000000800000008')            AT X'001445EC'
 001: ARG 009 (X'00000000000000080000000800000060')            AT X'001445F0'
```

Figure 43. Example CEDF output on exit from an MQSET call (hexadecimal)

```
 STATUS:  ABOUT TO EXECUTE COMMAND
 CALL TO RESOURCE MANAGER MQM
  001: ARG 000 ('................')
  001: ARG 001 ('................')
  001: ARG 002 ('.......... .e...')
  001: ARG 003 ('................')
  001: ARG 004 ('.......... .e...')
  001: ARG 005 ('................')
  001: ARG 006 ('............... .')
  001: ARG 007 ('................')
  001: ARG 008 ('................')
  001: ARG 009 ('...............-')
```

Figure 44. Example CEDF output on entry to an MQSET call (character)

```
STATUS:  COMMAND EXECUTION COMPLETE
CALL TO RESOURCE MANAGER MQM
 001: ARG 000 ('................')
 001: ARG 001 ('................')
 001: ARG 002 ('.......... .e...')
 001: ARG 003 ('................')
 001: ARG 004 ('.......... .e...')
 001: ARG 005 ('................')
 001: ARG 006 ('............... .')
 001: ARG 007 ('................')
 001: ARG 008 ('................')
 001: ARG 009 ('...............-')
```

Figure 45. Example CEDF output on exit from an MQSET call (character)

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-k,u
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England
SO21 2JN.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

COPYRIGHT LICENSE:

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

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

C/MVS	CICS	DB2
DFS	Extended Services	FFST
IBM	IBMLink	IMS
IMS/ESA	Language Environment	MQSeries
MVS	OS/390	RACF
RETAIN	RMF	VTAM
WebSphere	z/OS	

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

Index

Numerics

0Cx abend 21
5C6 abend
  associated reason codes 43
  diagnostic information 43
6C6 abend
  associated reason codes 43
  diagnostic information 43

A

abend
  0C4 8, 21
  0C7 21
  5C6 42
  6C6 42
  AICA 26
  ASRA 21
  CICS 44
  IMS 44
  internal error 42
  introduction 10
  no dump taken 82
  program 21
  severe error 42
  subsystem action 42
  z/OS 44
abend code
  description 42
  in dump title 61
ABEND keyword 77, 82
abnormal termination 42
address space
  channel initiator, dumping 49
  display list of active 53
  finding the identifier 55
  in dump formatting 53
  WebSphere MQ, dumping 49
AMQ message 86
analyzing dumps 61
APAR 8, 75
  definition 94
  number 97
  raising 97
application
  running slowly 16
  stopped 17
application design, performance considerations 31
application programming errors, examples 8
applying the fix 97
ASID in dump title 62
ASRA abend 21
ATB message 86
ATR message 86

B

batch
  abend 11, 21
  application loop 26
  wait 23
buffer pool size 29

C

CBC message 86
CEDF
  example output 107
  introduction 47
CEE message 86
CEMT INQ TASK 24
CFRM
  activation 33
  policy data set 33
change team 94
changed application 5
channel
  running 35
channel initiator dump
  formatting 59
channel initiator trace 65, 72
channel initiator, checking 17
channel problems 35
CICS
  abend 11, 21
  abend code 44
  application loop 26
  execution diagnostic facility 47
    example output 107
  performance considerations 30
  transaction wait 24
CICS adapter
  trace 74
  trace entries 103
CICS region
  CPU activity 26
CLASS, specifying 66
cluster
  finding a message 37
  message not on queue 33
  queue 15
CMQ modules 44
code, return 3
collecting documentation 95
command
  no response from 11
  problems 4
  to take a dump 49
command server 12
commands
  DISPLAY CONN 47
completion code in dump title 61
component identifier
  in dump title 61
  list of 101
component in dump title 61
component-identifier keyword 81
concurrent threads, limiting 30
content of dump 61
control blocks, display 53
CorrelId 34
CorrelId parameter 12
correlid, performance considerations 31
corrupted message 38
Coupling Facility structure
  accessibility 14
  full 26, 27
Coupling Facility structure dump
  formatting 60
CPU activity
  WebSphere MQ 25
CSECT
  in dump title 61
  keyword 84
  offset in dump title 62
CSECT keyword 76
CSMT log 24
CSQ message 3
CSQV086E message 83
CSQWDMP statement 56
CSQYASCP, 0C4 abend during startup 8
CSV message 86
CURDEPTH
  DISPLAY 12
  MAXDEPTH 13
  value of 37

D

D RRS 25
DAE (dump analysis and elimination) 64
data sets, distribution of 29
DB2
  abend 21
  wait 25
DB2 DISPLAY THREAD (*) 25
DB2 tables
  CSQ45STB job 96
dead-letter queue 12
  message header 27, 36
debugging
  common programming errors 8
  diagnostic aids 41
  preliminary checks 2
describing the problem 78
DFH message 86
DFS message 86
diagnostic aids
  channel initiator trace 65
  dumps 48
  GTF trace 65
  IBM internal trace 65
  introduction 41
  line trace 65
  SYS1.LOGREC records 64
  trace 65

diagnostic aids *(continued)*
user parameter trace 65
diagnostic information 43
display
dump title 52
queue 13, 46
system status 17
DISPLAY CF 27
DISPLAY CHSTATUS
BYTSSENT keyword 7
DISPLAY CMDSERV 12
DISPLAY CONN command 47
DISPLAY DQM 17, 18
DISPLAY QLOCAL 7
DISPLAY QMGR COMMANDQ 13
DISPLAY QUEUE 37
CURDEPTH attribute 12
MAXDEPTH attribute 12
DISPLAY THREAD 17, 25
distributed queuing 27
distributed queuing, message not on queue 35
DMQ message 86
DMQ modules 44
DOC keyword 76, 77, 88
documentation
problems 88
required for a PMR 95
sending PMR documentation 96
useful in problem determination 46
DSN message 86
dump
analyzing 61
content 61
display 53
format 53
using line mode IPCS 55
using the CSQWDMP statement 56
using the panels 51
managing the inventory 52
not taken for an abend 82
options 49
printing 60
processing
using IPCS in batch 60
using line mode IPCS 55
using the CSQWDMP statement 56
using the dump display panels 51
selecting 52, 55
summary portion 49
suppression 64
SYS1.LOGREC data 53
taking 49
title 61
using the z/OS dump command 49
dump analysis and elimination (DAE) 64
dump inventory, managing 52
dumps
address space 11
transaction 11

E

EDC message 86
error message
finding explanation 3
unexpected 10
error messages
non-WebSphere MQ 10
error, user data 41
event identifier 65
examining the problem in-depth 9
example output
CEDF 107
SYSUDUMP 62
EZA message 86
EZB message 86
EZY message 86

F

failure keywords 77
fix, applying and testing 97
formatting
channel initiator dump 59
Coupling Facility structure dump 60
formatting a dump
using line mode IPCS 55
using the CSQWDMP statement 56
using the panels 51
free keyword format 77
frequent I/O 28
FRR keyword 85

G

grouped messages 36
GTF
format identifier 68
formatting 67
identifying WebSphere MQ control blocks 68
if no data is produced 68
interpreting 68
specifying the job name 65
starting 65
user parameter trace 65
USRP option 65
GTFTRACE command 67

H

HALT keyword 77

I

I/O, frequent 28
IBM
software support 91
software support database, searching 75
support center 75
change team 94
dealing with 91
Development Support Group 94
ordering a specific PTF 98
what they need to know 92
IBM *(continued)*
support center *(continued)*
when to contact 91
IBM internal trace 65
IBM message 86
ICH message 86
IDC message 86
identifier
component 101
resource manager 101
IEA message 86
IEA911E message 82
IEC message 86
IEE message 86
IEF message 86
IKJ message 86
IMQ modules 44
IMS
abend 11, 21
abend code 44
CPU activity 26
loop 26
message flow 38
IMS bridge, message not arriving 37
incident number 93
incorrect output 9
INCORROUT keyword 77, 89
index, queue 31
Information/Access 75
Information/System 75
initialization procedure 4
INTEG keyword 77
intermittent problems 7
IPCS subcommands 58
IPCS VERBEXIT CSQXDPRD keywords 59
IPPROCS attribute 13
IST message 86
IWM message 86
IXC message 86

J

job name, specifying for GTF 65

K

keyword
building a string 78
component-identifier 81
CSECT 84
modifier
load module 84
recovery routine 85
prefix 77
release level 81
selecting 78
symptom-to-keyword cross-reference 99
type-of-failure
ABEND 82
determining 81
DOC 88
INCORROUT 89
LOOP 85
MSG 85

keyword *(continued)*
type-of-failure *(continued)*
PERFM 87
WAIT 85
keyword format
free 77
structured database 77
z/OS 77

L

line trace 65
load module
in dump title 61
modifier keyword 84
Load Module modifier keyword 76
log buffer pools 28
long-running unit of work 14
loop
batch application 26
causes 23
CICS application 26
distinguishing from a wait 22
IMS region 26
TSO application 26
WebSphere MQ 26
LOOP keyword 77, 85
LU 6.2 connection 35

M

manuals
problems 88
MAXMSGL 33
message
contains unexpected information 38
CSQ 3
CSQV086E 83
error 3
grouping 36
IEA911E 82
incorrect queue 38
not appearing on queue 33
not on queue
cluster queue 37
IMS bridge 37
trigger information 38
variable length, performance considerations 32
where to find more information 86
message length, performance considerations 31
message persistence, performance considerations 31
modifier keyword
load module 84
recovery routine 85
module, in dump title 61
MQPMO structure 37
MQPUT and MQPUT1, performance considerations 33
MSG keyword 85
MsgId 34
MsgId parameter 12
msgid, performance considerations 31
multiple transactions 34

N

naming convention 44
network problems 6
new application 5
nonpersistent messages 36
NPMSPEED attribute 36

O

operator command, no response from 11
OPPROCS attribute 14
ordering a specific PTF 98
OTMA
connection to WebSphere MQ 37
outage
messages in doubt 35

P

page set full 26
panels, dump display 51
PERFM keyword 77, 87
performance considerations 27
performance degradation 16
persistent messages 34
PING CHANNEL command 7
prefix keyword 77
preliminary checks 2
primary keywords 93
printing dumps 60
problem
cause of 2
management record 93
reporting sheet 91
tracking 91
processing a dump
using IPCS in batch 60
using line mode IPCS 55
using the CSQWDMP statement 56
using the panels 51
product support services software support 91
program abend 21
program check 41
program error 41
queue manager detected 41
user-detected 41
programming errors, examples 8
PSW, in dump title 62
PTF 8, 75
definition 94
ordering 98
publications problem 88

Q

queue
cluster 15
display 13
distribution of 29
full 26, 27
index 31
remote 15
queue full 33
queue information, displaying 46
queue manager
CPU activity 26
queue manager detected errors 41
queue-sharing environment
error messages 10

R

raising an APAR 97
reason code
associated with subsystem abend 43
in dump title 61
recovery actions 41
recovery routine keyword 85
recovery routine modifier keyword 76
release in dump title 61
release-level keyword 76, 81
remote queue
initial checks 15
message not on queue 35
remote system
reply message 36
reporting a problem 91
resident trace table, display 53
resolving a problem 97
resource manager formatting keywords 56
resource manager identifier, list of 101
RETAIN 93, 94
searching 75
symptoms 77
return codes 3
changed application 5
RMF 18
RRS
abend 21
active 25
RRS, active 14

S

sample problem reporting sheet 91
save area trace report, display 54
SDB (structured database) keyword format 77
SDB format keywords 99
search argument
process 75
varying 76
selecting a dump 52, 55
sending documentation 96
severity level 93
shared queue
message not on queue 33
shared queues 15
initial checks 15
snap dump 49, 63
software support 91
software support database, searching 75
stand-alone dump 49
START CMDSERV 12
starting the GTF 65
starting the trace 66
STOP QMGR 25
stopping the GTF 67

structured database (SDB) keyword
format 77
SUBSYS= parameter 53, 56, 58
subsystem name
finding 54
in dump formatting 53
in dump title 61
subsystem termination 42, 43
SUMDUMP= parameter 53, 58
summary dump in dump formatting 53
summary portion of a dump 49
suppressing dumps 64
SVC dump 43, 49
printing 60
processing
using IPCS in batch 60
using line mode IPCS 55
using the CSQWDMP statement 56
using the dump display panels 51
suppression 64
title 61
title, variation with PSW and ASID 62
symptom keywords 78
symptom string
display 53
introduction 45
symptom-to-keyword cross-reference 99
syncpoint 12, 34
syncpoint, performance considerations 32
SYS1.DUMPxx data set 52
SYS1.LOGREC
analyzing 64
data 53
finding the applicable information 64
introduction 43
system
changes 4
displaying status 17
running slowly 16
stopped 17
system abend completion code 42
system diagnostic work area, display 53
system loading
time of day 7
System SSL
trace 74
SYSUDUMP
introduction 49
sample 62

T

TCP/IP connection 35
termination, abnormal 42
testing the fix 97
trace
channel initiator 72
CICS adapter 74
entries for CICS adapter 103
format identifier 68
formatting 67
identifying WebSphere MQ control blocks 68
trace *(continued)*
if no data is produced 68
interpreting 68
performance considerations 30
specifying the CLASS 66
starting 66
System SSL 74
user parameters 65
z/OS 74
trace table, display 53
trace types 65
transmission queue 35
trigger
definition 27
monitor 34
options 34
process 34
TSO
abend 11, 21
application loop 26
wait 23
type-of-failure keyword
ABEND 82
determining 81
DOC 88
INCORROUT 89
LOOP 85
MSG 85
PERFM 87
symptom-to-keyword cross-reference 99
WAIT 85

U

unit of work, long-running 14
user data related errors 41
user identifier, in dump title 61
user parameter trace 65
user-detected program errors 41
userid, in dump title 61

V

variable recording area 43
variable recording area, display 53
VERBEXITs 58

W

wait
batch 23
causes 22
CICS transaction 24
DB2 25
distinguishing from a loop 22
TSO 23
WebSphere MQ 25
WAIT keyword 77, 85
WaitInterval parameter 12
WebSphere MQ
loop 26
running slowly 16
wait 25
WebSphere MQ DB2 tables
CSQ45STB job 96
WebSphere MQ-IMS bridge, message not arriving 37

X

XES
abend 21

Z

z/OS
abend 44
trace 74
using the dump command 49
z/OS under stress 28

Sending your comments to IBM

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

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

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

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

When you send comments to IBM , you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

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

- By mail, to this address:

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

Whichever method you use, ensure that you include:

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

GC34-6944-01

Spine information:

WebSphere MQ for z/OS Problem Determination Guide Version 7.0