

MQSeries® for VSE/ESA



System Management Guide

Version 2 Release 1

MQSeries® for VSE/ESA



System Management Guide

Version 2 Release 1

Note!

Before using this information and the product it supports, be sure to read the general information under Appendix I, "Notices" on page 219.

First edition (October 1998)

This edition applies to the following product:

- MQSeries for VSE/ESA Version 2 Release 1

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

Order publications through your IBM® representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

At the back of this publication is a page titled "Sending your comments to IBM". If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories, Information Development,
Mail Point 095, Hursley Park, Winchester, Hampshire, England,
SO21 2JN

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1995, 1998. All rights reserved.**

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

About this book	ix
Who this book is for	ix
What you need to know to understand this book	ix
How to use this book	ix
MQSeries publications	x
MQSeries cross-platform publications	x
MQSeries platform-specific publications	xii
MQSeries Level 1 product publications	xiii
Softcopy books	xiv
MQSeries information available on the Internet	xv
Related publications	xv
New function in MQSeries for VSE/ESA Version 2.1	xvii
Chapter 1. Introduction	1
MQSeries and message queuing	1
Messages and queues	1
Objects	3
Clients and servers	6
MQSeries and CICS	7
Chapter 2. Installation	9
Contents of the library tape	9
Installing MQSeries for VSE/ESA – all users	10
Procedures for new users	12
Starting MQSeries	15
MQSeries installation verification test	19
Post installation verification test CICS modifications	24
Migration procedures for existing users	24
Chapter 3. Configuring network communications	29
MQSeries system definitions required	29
MQSeries for VSE/ESA configuration guidelines	33
Chapter 4. System operation	41
MQSeries master terminal displays	41
General panel layout	42
MQSeries master terminal (MQMT) – main menu	42
Configuration functions	44
Operations functions	66
Monitoring functions	75
Browse function	80
Communications process	81
Viewing error logs	81
Chapter 5. MQSeries for VSE/ESA utility functions	83
System Administration Control Interface	83
Background batch modules	86
Using the batch interface	88
VSAM file maintenance	90

Chapter 6. Problem determination	95
MQSeries setup and local queue operation	95
Network problems	96
Applications	98
Other areas of investigation	102
Application design considerations	105
Incorrect output	106
System log	108
Dead-letter queues	109
Using MQSeries trace	109
Problem determination with clients	109
Appendix A. MQSeries for VSE/ESA at a glance	111
Appendix B. CICS control table definitions	113
Sample file control table entries	113
Sample destination control table entry	114
Sample JCL to process MQPUTIL	115
Sample JCL file definition for CICS deck	116
Sample JCL to create CICS CSD group	116
Appendix C. Application Programming Reference	119
Structure data types	119
MQI calls	126
Attributes of MQSeries objects	136
Reason codes	136
Appendix D. Application Programming Guidance	137
Supporting application programs that use the MQI	137
Sample source code provided	138
Application design guidelines	138
Appendix E. Sample programs	145
Sample program TTPTST2.Z	145
Sample program TTPTST3.Z	161
Sample program MQPECHO.Z	173
Appendix F. Example configuration - MQSeries for VSE/ESA Version 2.1.0	185
Configuration parameters for an LU 6.2 connection	185
Establishing an LU 6.2 connection	189
Establishing a TCP/IP connection	190
MQSeries for VSE/ESA configuration	190
Appendix G. MQSeries clients	199
Client support	199
Security considerations	199
Client code-page conversion tables	200
Appendix H. System messages	201
API system messages	201
MQSeries message definitions	202
MQSeries message codes	203
Console Messages	217

Appendix I. Notices	219
Glossary of terms and abbreviations	221
Index	229

Contents

Figures

1.	Default screen	16
2.	Configuration main menu	18
3.	TPTTST2 screen	19
4.	Monitor queues screen	20
5.	Browse Queue Records screen – status written	21
6.	Browse Queue Records screen – status deleted	22
7.	Definitions in CICS using RDO for parallel session partner LU	30
8.	Definitions in CICS for single-session capable partner LU.	31
9.	Definitions in CICS singles-session capable LU	31
10.	Outline MQSeries channel definition	35
11.	Display screen relationships	41
12.	General panel layout	42
13.	Master terminal main menu	43
14.	Configuration main menu	44
15.	System queue manager information	46
16.	Queue main options screen	48
17.	Local queue definition	49
18.	Local queue extended definition	51
19.	Automatic reorganization	54
20.	Remote queue definition	55
21.	Alias queue definition	56
22.	Alias queue manager definition	57
23.	Alias queue reply definition	58
24.	Object list screen	60
25.	Channel record	61
26.	Channel list	64
27.	Global system definition display	65
28.	Operations main menu	66
29.	Start / Stop queue control screen	67
30.	Open / Close Channel	69
31.	Reset channel message sequence	70
32.	Initialization of system	72
33.	Maintain Queue Message Records	73
34.	Monitor Main Menu	75
35.	Monitor queues	75
36.	Monitor Queues - detail	77
37.	Monitor channel definitions	78
38.	Monitor channel definitions - detail	79
39.	Browse Queue Records	80
40.	API monitor	100
41.	API monitor - browse	101
42.	API monitor - hexadecimal format	101
43.	Queues, messages, and applications	137
44.	Test System Programs 3 - start	161
45.	Channel configuration panel	197

Tables

1. Object Characteristics of Connection	32
2. CEMT I CONN display output.	32
3. CEDA V SESS display parameter settings	32
4. Example queue manager configuration	39
5. Example channel configuration	40
6. Example queue configuration	40
7. MQPUTIL program general syntax	86
8. Configuration worksheet for VSE/ESA using APPC	186
9. Configuration worksheet for MQSeries for VSE/ESA	190

About this book

MQSeries for VSE/ESA Version 2.1—referred to in this book as MQSeries for VSE/ESA or simply MQSeries, as the context permits—is part of the MQSeries family of products. These products provide application programming services that enable application programs to communicate with each other using *message queues*. This form of communication is referred to as *commercial messaging*. The applications involved can exist on different nodes on a wide variety of machine and operating system types. They use a common application programming interface, called the Message Queuing Interface or MQI, so that programs developed on one platform can be readily transferred to another.

This book describes the system administration aspects of MQSeries for VSE/ESA Version 2.1 and the services it provides to support commercial messaging in an VSE/ESA environment. This includes managing the queues that applications use to receive their messages, and ensuring that applications have access to the queues that they require.

Who this book is for

Primarily, this book is for system administrators, and system programmers who manage the configuration and administration tasks for MQSeries. It is also useful to application programmers who must have some understanding of MQSeries administration tasks.

What you need to know to understand this book

To use this book, you should have a good understanding of the VSE/ESA operating system, and utilities associated with it. You do not need to have worked with message queuing products before, but you should have an understanding of the basic concepts of message queuing.

How to use this book

Read Chapter 1, “Introduction” on page 1 first for an understanding of MQSeries for VSE/ESA.

The body of this book contains:

- Chapter 2, “Installation” on page 9
- Chapter 3, “Configuring network communications” on page 29
- Chapter 4, “System operation” on page 41
- Chapter 5, “MQSeries for VSE/ESA utility functions” on page 83
- Chapter 6, “Problem determination” on page 95

At the back of the book there are some appendixes giving information (which will be incorporated in the appropriate MQSeries books at the next opportunity) on the following topics:

- Appendix A, “MQSeries for VSE/ESA at a glance” on page 111
- Appendix B, “CICS control table definitions” on page 113
- Appendix C, “Application Programming Reference” on page 119

MQSeries publications

- Appendix D, “Application Programming Guidance” on page 137
- Appendix E, “Sample programs” on page 145
- Appendix F, “Example configuration - MQSeries for VSE/ESA Version 2.1.0” on page 185
- Appendix G, “MQSeries clients” on page 199
- Appendix H, “System messages” on page 201

MQSeries publications

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

MQSeries cross-platform publications

Most of these publications, which are sometimes referred to as the MQSeries “family” books, apply to all MQSeries Level 2 products. The latest MQSeries Level 2 products are:

- MQSeries for AIX V5.0
- MQSeries for AS/400 V4R2M1
- MQSeries for AT&T GIS UNIX V2.2
- MQSeries for Digital OpenVMS V2.2
- MQSeries for HP-UX V5.0
- MQSeries for MVS/ESA V1.2
- MQSeries for OS/2 Warp V5.0
- MQSeries for SINIX and DC/OSx V2.2
- MQSeries for Sun Solaris V5.0
- MQSeries for Tandem NonStop Kernel V2.2
- MQSeries for Windows V2.0
- MQSeries for Windows V2.1
- MQSeries for Windows NT V5.0

Any exceptions to this general rule are indicated. (Publications that support the MQSeries Level 1 products are listed in “MQSeries Level 1 product publications” on page xiii. For a functional comparison of the Level 1 and Level 2 MQSeries products, see the *MQSeries Planning Guide*.)

MQSeries Brochure

The *MQSeries Brochure*, G511-1908, gives a brief introduction to the benefits of MQSeries. It is intended to support the purchasing decision, and describes some authentic customer use of MQSeries.

MQSeries: An Introduction to Messaging and Queuing

MQSeries: An Introduction to Messaging and Queuing, GC33-0805, describes briefly what MQSeries is, how it works, and how it can solve some classic interoperability problems. This book is intended for a more technical audience than the *MQSeries Brochure*.

MQSeries Planning Guide

The *MQSeries Planning Guide*, GC33-1349, describes some key MQSeries concepts, identifies items that need to be considered before MQSeries is installed, including storage requirements, backup and recovery, security, and migration from earlier releases, and specifies hardware and software requirements for every MQSeries platform.

MQSeries Intercommunication

The *MQSeries Intercommunication* book, SC33-1872, defines the concepts of distributed queuing and explains how to set up a distributed queuing network in a variety of MQSeries environments. In particular, it demonstrates how to (1) configure communications to and from a representative sample of MQSeries products, (2) create required MQSeries objects, and (3) create and configure MQSeries channels. The use of channel exits is also described.

MQSeries Clients

The *MQSeries Clients* book, GC33-1632, describes how to install, configure, use, and manage MQSeries client systems.

MQSeries System Administration

The *MQSeries System Administration* book, SC33-1873, supports day-to-day management of local and remote MQSeries objects. It includes topics such as security, recovery and restart, transactional support, problem determination, the dead-letter queue handler, and the MQSeries links for Lotus Notes®. It also includes the syntax of the MQSeries control commands.

This book applies to the following MQSeries products only:

- MQSeries for AIX V5.0
- MQSeries for HP-UX V5.0
- MQSeries for OS/2 Warp V5.0
- MQSeries for Sun Solaris V5.0
- MQSeries for Windows NT V5.0

MQSeries Command Reference

The *MQSeries Command Reference*, SC33-1369, contains the syntax of the MQSC commands, which are used by MQSeries system operators and administrators to manage MQSeries objects.

MQSeries Programmable System Management

The *MQSeries Programmable System Management* book, SC33-1482, provides both reference and guidance information for users of MQSeries events, programmable command formats (PCFs), and installable services.

MQSeries Messages

The *MQSeries Messages* book, GC33-1876, which describes “AMQ” messages issued by MQSeries, applies to these MQSeries products only:

- MQSeries for AIX V5.0
- MQSeries for HP-UX V5.0
- MQSeries for OS/2 Warp V5.0
- MQSeries for Sun Solaris V5.0
- MQSeries for Windows NT V5.0
- MQSeries for Windows V2.0
- MQSeries for Windows V2.1

This book is available in softcopy only.

MQSeries Application Programming Guide

The *MQSeries Application Programming Guide*, SC33-0807, provides guidance information for users of the message queue interface (MQI). It describes how to design, write, and build an MQSeries application. It also includes full descriptions of the sample programs supplied with MQSeries.

MQSeries publications

MQSeries Application Programming Reference

The *MQSeries Application Programming Reference*, SC33-1673, provides comprehensive reference information for users of the MQI. It includes: data-type descriptions; MQI call syntax; attributes of MQSeries objects; return codes; constants; and code-page conversion tables.

MQSeries Application Programming Reference Summary

The *MQSeries Application Programming Reference Summary*, SX33-6095, summarizes the information in the *MQSeries Application Programming Reference* manual.

MQSeries Using C++

MQSeries Using C++, SC33-1877, provides both guidance and reference information for users of the MQSeries C++ programming-language binding to the MQI. MQSeries C++ is supported by V5.0 of MQSeries for AIX, HP-UX, OS/2 Warp, Sun Solaris, and Windows NT, and by MQSeries clients supplied with those products and installed in the following environments:

- AIX®
- HP-UX
- OS/2®
- Sun Solaris
- Windows NT
- Windows 3.1
- Windows 95

MQSeries C++ is also supported by MQSeries for AS/400 V4R2.

MQSeries platform-specific publications

Each MQSeries product is documented in at least one platform-specific publication, in addition to the MQSeries family books.

MQSeries for AIX

MQSeries for AIX V5.0 Quick Beginnings, GC33-1867

MQSeries for AS/400®

MQSeries for AS/400 Version 4 Release 2 Modification 1 Administration Guide, GC33-1956

MQSeries for AS/400 Version 4 Release 2 Application Programming Reference (RPG), SC33-1957

MQSeries for AT&T GIS UNIX®

MQSeries for AT&T GIS UNIX Version 2.2 System Management Guide, SC33-1642

MQSeries for Digital OpenVMS

MQSeries for Digital OpenVMS Version 2.2 System Management Guide, GC33-1791

MQSeries for HP-UX

MQSeries for HP-UX V5.0 Quick Beginnings, GC33-1869

MQSeries for MVS/ESA™

MQSeries for MVS/ESA Version 1 Release 2 Licensed Program Specifications, GC33-1350

MQSeries for MVS/ESA Version 1 Release 2 Program Directory

MQSeries for MVS/ESA Version 1 Release 2 System Management Guide,
SC33-0806

MQSeries for MVS/ESA Version 1 Release 2 Messages and Codes,
GC33-0819

MQSeries for MVS/ESA Version 1 Release 2 Problem Determination Guide,
GC33-0808

MQSeries for OS/2 Warp

MQSeries for OS/2 Warp V5.0 Quick Beginnings, GC33-1868

MQSeries link for R/3

MQSeries link for R/3 Version 1.2 User's Guide, GC33-1934

MQSeries for SINIX and DC/OSx

MQSeries for SINIX and DC/OSx Version 2.2 System Management Guide,
GC33-1768

MQSeries for Sun Solaris

MQSeries for Sun Solaris V5.0 Quick Beginnings, GC33-1870

MQSeries for Tandem NonStop Kernel

*MQSeries for Tandem NonStop Kernel Version 2.2 System Management
Guide*, GC33-1893

MQSeries for Windows®

MQSeries for Windows Version 2.0 User's Guide, GC33-1822

MQSeries for Windows Version 2.1 User's Guide, GC33-1965

MQSeries for Windows NT®

MQSeries for Windows NT V5.0 Quick Beginnings, GC33-1871

MQSeries Level 1 product publications

For information about the MQSeries Level 1 products, see the following publications:

MQSeries: Concepts and Architecture, GC33-1141

*MQSeries Version 1 Products for UNIX Operating Systems Messages and
Codes*, SC33-1754

MQSeries for UnixWare Version 1.4.1 User's Guide, SC33-1379

Softcopy books

Most of the MQSeries books are supplied in both hardcopy and softcopy formats.

BookManager™ format

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

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

PostScript format

The MQSeries library is provided in PostScript (.PS) format with many MQSeries products, including all MQSeries V5.0 products. Books in PostScript format can be printed on a PostScript printer or viewed with a suitable viewer.

HTML format

The MQSeries documentation is provided in HTML format with these MQSeries products:

- MQSeries for AIX V5.0
- MQSeries for HP-UX V5.0
- MQSeries for OS/2 Warp V5.0
- MQSeries for Sun Solaris V5.0
- MQSeries for Windows NT V5.0

The MQSeries books are also available from the MQSeries product family Web site:

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

Information Presentation Facility (IPF) format

In the OS/2 environment, the MQSeries documentation is supplied in IBM IPF format on the MQSeries product CD-ROM.

Windows Help format

The *MQSeries for Windows User's Guide* is provided in Windows Help format with MQSeries for Windows Version 2.0 and MQSeries for Windows Version 2.1.

MQSeries information available on the Internet

MQSeries web site

The MQSeries product family Web site is at:

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

By following links from this Web site you can:

- Obtain latest information about the MQSeries product family.
- Access the MQSeries books in HTML format.
- Download MQSeries SupportPacs.

Related publications

CICS/VSE® V2.3 Resource Definition (Macro), SC33-0709

CICS/VSE V2.3 System Definition and Operations Guide, SC33-0706

Language Environment for VSE/ESA, C Run-Time Programming Guide,
SC33-6688

TCP/IP for VSE/ESA User's Guide, SC33-6601

Related publications

New function in MQSeries for VSE/ESA Version 2.1

This section lists the major additions to the product for this release; these additions include:

- Automatic file reorganization
- Maximum message size of 4 MB
- Message priority on MQPUT
- Message batching for channels
- Message identifier and correlation identifier support for MQGET
- Nonpersistent message support
- Channel communication through TCP/IP
- Default code page support using a user-specified CCSID
- Support for the C and PL/I programming languages
- Confirm on arrival and delivery message reports
- Support for client attach through TCP/IP
- Extra diagnostic information when communications fail

New function

Chapter 1. Introduction

This chapter introduces MQSeries for VSE/ESA from an administrator's perspective, and describes the basic concepts of MQSeries and messaging.

MQSeries and message queuing

MQSeries lets VSE/ESA applications use message queuing to participate in message-driven processing. Applications can communicate across different platforms by using the appropriate message queuing software products. For example, VSE/ESA and MVS/ESA applications can communicate through MQSeries for VSE/ESA and MQSeries for MVS/ESA respectively. The applications are shielded from the mechanics of the underlying communications.

MQSeries products implement a common application programming interface (message queue interface or MQI) whatever platform the applications are run on. This makes it easier to port applications from one platform to another.

The MQI is described in detail in the *MQSeries Application Programming Reference* manual.

Time-independent applications

With message queuing, the exchange of messages between the sending and receiving programs is time independent. This means that the sending and receiving applications are decoupled so that the sender can continue processing without having to wait for the receiver to acknowledge the receipt of the message. In fact, the target application does not even have to be running when the message is sent. It can retrieve the message after it is started.

Message-driven processing

Applications can be automatically started by messages arriving on a queue using a mechanism known as *triggering*. If necessary, the applications can be stopped when the message or messages have been processed.

Messages and queues

Messages and queues are the basic components of a message queuing system.

What messages are

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

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

Messages and queues

The format of the message descriptor is defined by MQSeries for VSE/ESA. For a complete description of the message descriptor, see the *MQSeries Application Programming Reference* manual.

Message lengths

In MQSeries for VSE/ESA, the maximum message length is 4 MB (where 1 MB equals 1 048 576 bytes). In practice, the message length may be limited by:

- The maximum message length defined for the receiving queue.
- The maximum message length defined for the queue manager.
- The maximum message length defined by either the sending or receiving application.
- The amount of storage available for the message.

This parameter is extremely important for MQSeries for VSE/ESA. The storage will be used from the CICS® partition in which the queue manager is active.

It may take several messages to send all the information that an application requires.

What queues are

A *queue* is a data structure that stores zero or more messages. The messages may be put on the queue by applications or by a queue manager as part of its normal operation.

Each queue belongs to a *queue manager*, which is responsible for maintaining it. The queue manager puts the messages it receives on the appropriate queues.

Applications send and receive messages using MQI calls. For example, one application can put a message on a queue, and another application can retrieve the message from the same queue. The sending application opens the queue for put operations by making an MQOPEN call. Then it issues an MQPUT call to put the message onto that queue. When the receiving application opens the same queue for gets, it can retrieve the message from the queue by issuing an MQGET call.

For more information about MQI calls, see the *MQSeries Application Programming Reference* manual.

MQSeries for VSE/ESA supports only *predefined queues*, which are those created by an administrator using the appropriate command set, for example, those defined using the MQSeries Master Terminal (MQMT) utility. Predefined queues are permanent; they exist independently of the applications that use them and survive MQSeries for VSE/ESA restarts.

Retrieving messages from queues

In MQSeries for VSE/ESA, suitably authorized applications can retrieve messages from a queue according to these retrieval algorithms:

- First-in-first-out (FIFO).
- A program request for a specific message, identified by a message identifier or correlation identifier.

The MQGET request from the application determines the method used.

Objects

Many of the tasks described in this book involve manipulating MQSeries *objects*. In MQSeries for VSE/ESA, there are three different types of object:

- Queue managers; see “MQSeries queue managers” on page 4.
- Queues; see “MQSeries queues” on page 4.
- Channels; see “Channels” on page 6.

Object names

Each instance of a queue manager is known by its name. This name must be unique within the network of interconnected queue managers, so that one queue manager can unambiguously identify the target queue manager to which any given message should be sent.

For the other types of object, each object has a name associated with it and can be referenced in MQSeries for VSE/ESA by that name. These names must be unique within one queue manager and object type. For example, you can have a queue and a process with the same name, but you cannot have two queues with the same name.

In MQSeries, names can have a maximum of 48 characters, with the exception of *channel names*, which have a maximum of 20 characters.

Managing objects

MQSeries provides commands for creating, altering, displaying, and deleting objects through the panel driven MQSeries Master Terminal (MQMT) system administration transaction; see “MQSeries master terminal (MQMT) – main menu” on page 42 for further details.

Note: Default object definitions are not supplied with MQSeries for VSE/ESA.

You can perform some limited administration, for example, the starting and stopping of queues and channels, by using the MQCL transaction. See Chapter 5, “MQSeries for VSE/ESA utility functions” on page 83 for further details.

Local and remote administration

Local administration means carrying out administration tasks on any queue managers you have defined on your local system. You can access other systems, for example through TCP/IP, and carry out administration there. In MQSeries, you can consider this as local administration because no channels are involved, that is, the communication is managed by the operating system.

Object attributes

The properties of an object are defined by its attributes. Some you can specify, others you can only view. For example, the maximum message length that a queue can accommodate is defined by its *MaxMsgLength* attribute (see Figure 18 on page 51). You can specify this attribute when you create a queue.

In MQSeries for VSE/ESA, there are two ways of accessing an attribute:

- Using the MQMT transaction, described in “MQSeries master terminal (MQMT) – main menu” on page 42

Objects

- Using the MQINQ function call, described in “MQINQ – Inquire about object attributes” on page 129

MQSeries queue managers

A queue manager provides queuing services to applications, and manages the queues that belong to it. There can only be one queue manager on each MQSeries for VSE/ESA system. It ensures that:

- Object attributes are changed according to the commands received.
- Special events such as trigger events are generated when the appropriate conditions are met.
- Messages are put on the correct queue, as requested by the application making the MQPUT call. The application is informed if this cannot be done, and an appropriate reason code is given.

Each queue belongs to a single queue manager and is said to be a *local queue* to that queue manager. The queue manager to which an application is connected is said to be the local queue manager for that application. For the application, the queues that belong to its local queue manager are local queues.

A *remote queue* is simply a queue that belongs to another queue manager. A *remote queue manager* is any queue manager other than the local queue manager. A remote queue manager exists on a remote machine across the network.

MQI calls

A queue manager object may be used in some MQI calls. For example, you can inquire about the attributes of the queue manager object using the MQI call MQINQ.

Note: You cannot put messages on a queue manager object; messages are always put on queue objects, not on queue manager objects.

MQSeries queues

Queues are defined to MQSeries using the appropriate MQMT transaction. The transaction specifies the type of queue and its attributes. For example, a local queue object has attributes that specify what happens when applications reference that queue in MQI calls. Examples of attributes are:

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

Using queue objects

In MQSeries for VSE/ESA, there are three types of queue object. Each type of object can be manipulated by the product commands and is associated with real queues in different ways:

1. A *local queue* object identifies a local queue belonging to the queue manager to which the application is connected. All queues are local queues in the sense that each queue belongs to a queue manager and, for that queue manager, the queue is a local queue.
2. A *remote queue object* identifies a queue belonging to another queue manager. This queue must be defined as a local queue to that queue manager. The information you specify when you define a remote queue object allows the local queue manager to find the remote queue manager, so that any messages destined for the remote queue go to the correct queue manager.

You must also define a transmission queue and channels between the queue managers, before applications can send messages to a queue on another queue manager.

3. An *alias queue object* allows applications to access a queue by referring to it indirectly in MQI calls. When an alias queue name is used in an MQI call, the name is resolved to the name of either a local or a remote queue at run time. This allows you to change the queues that applications use without changing the application in any way—you merely change the alias queue definition to reflect the name of the new queue to which the alias resolves.

An alias queue is not a queue, but an object that you can use to access another queue.

Specific local queues used by MQSeries

MQSeries uses some local queues for specific purposes related to its operation. You **must** define them before MQSeries can use them.

Application queues: A queue that is used by an application (through the MQI) is referred to as an *application queue*. This can be a local queue on the queue manager to which an application is linked, or it can be a remote queue that is owned by another queue manager.

Applications can put messages on local or remote queues. However, they can only get messages from a local queue.

Transmission queues: A *transmission queue* temporarily stores messages that are destined for a remote queue manager. You must define at least one transmission queue for each remote queue manager to which the local queue manager is to send messages directly. For information about the use of transmission queues in distributed queuing, see the *MQSeries Intercommunication* book.

Dead-letter queues: A *dead-letter queue* stores messages that cannot be routed to their correct destinations. This occurs when, for example, the destination queue is full. The supplied dead-letter queue is called SYSTEM.DEAD.LETTER.QUEUE. These queues are also referred to as undelivered-message queues on other platforms.

For distributed queuing, you should define a dead-letter queue for each queue manager.

Clients and servers

Event queues: MQSeries for VSE/ESA does not support instrumentation events.

Channels

Channels are objects that provide a communication path from one queue manager to another. Channels are used in distributed message queuing to move messages from one queue manager to another. They shield applications from the underlying communications protocols. The queue managers may exist on the same, or different, platforms. For queue managers to communicate with one another, you must define one channel object at the queue manager that is to send messages, and another, complementary one, at the queue manager that is to receive them.

For information on channels and how to use them, see the *MQSeries Intercommunication* book.

Clients and servers

MQSeries for VSE/ESA supports client-server configurations for MQSeries applications, and can act as a server to which all current MQSeries clients can connect.

Note: There is no VSE/ESA client.

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

An *MQI server* is a queue manager that provides queuing services to one or more clients. For VSE/ESA, there is one MQSeries process for each client connection.

All the MQSeries objects, for example queues, exist only on the queue manager machine, that is, on the MQI server machine. A server can support normal local MQSeries applications as well.

The difference between an MQI server and an ordinary queue manager is that a server has a dedicated communications link with each client. For more information about creating channels for clients and servers, see the *MQSeries Intercommunication* book.

MQSeries applications in a client-server environment

When linked to a server, client MQSeries applications can issue MQI calls in the same way as local applications. The client application issues an MQCONN call to connect to a specified queue manager. Any additional MQI calls that specify the connection handle returned from the connect request are then processed by this queue manager. You must link your applications to the appropriate client libraries. See the *MQSeries Application Programming Guide* for further information.

MQSeries and CICS

Note to users

MQSeries for VSE/ESA runs as a CICS task. Consequently, various features of the product are controlled by CICS itself.

These features include security and recovery.

Chapter 2. Installation

This chapter describes the procedure for installing MQSeries for VSE/ESA. It consists of the following sections:

1. "Contents of the library tape"
2. "Installing MQSeries for VSE/ESA – all users" on page 10
3. "Procedures for new users" on page 12
4. "Starting MQSeries" on page 15
5. "MQSeries installation verification test" on page 19
6. "Post installation verification test CICS modifications" on page 24
7. "Migration procedures for existing users" on page 24

Contents of the library tape

The distribution tape is in standard IBM MSHP format and is in V2 stacked format. The tape may include other IBM products and contains a sublibrary for "PRD2.MQSERIES".

This sublibrary contains:

- Copy books, used by your CICS applications whenever you intend to call the MQSeries Application Programming Interface (API).
- Object decks, called at linkedit time when you are building your own MQSeries applications (autolink).
- Phases, which are all compiled in COBOL for VSE with LE/VSE and linkedited with AMODE(31) and RMODE(ANY).
- Samples having member type Z. Some of these need to be modified for the VSE/POWER JECL statements, as follows:

```

* ** JOB to * $$ JOB
* ** LST to * $$ LST
* ** SLI to * $$ SLI
* ** E0J to * $$ E0J

```

The samples are:

MQJCONFIG.Z Creation of MQSeries configuration file
MQJSETUP.Z Creation of the setup file
MQJQUEUE.Z VSAM cluster definitions for MQSeries queues
MQJMIGR1.Z Migration of old configuration file (step 1)
MQJMIGR2.Z Migration of old configuration file (step 2)
MQJREORG.Z Batch job to reclaim space of deleted records
MQJUTILY.Z Various batch functions
MQJLABEL.Z Label definitions for the CICS start-up job
MQJCSD.Z Define CICS resources into the CICS CSD
MQCICDCT.Z Entry definitions for CICS DCT
MQCICFCT.Z Entry definitions for CICS FCT

MQUSERID.Z Sample assembler to allow a change of user identifier for MCA communications with remote AS/400 systems

Installing MQSeries for VSE/ESA – all users

To install the product, carry out the following procedure:

1. Decide the name of the :

- Target sublibrary

The target sublibrary can be the default supplied, “PRD2.MQSERIES”, or a name that you specify.

If you use the supplied default sublibrary, go to step 2 on page 11.

If you specify your own library, you must customize the JCL listed in step 1b.

- VSAM catalog into which the product is to be installed

a. Create a VSAM user catalog.

You are recommended to use the Interactive Interface Dialogs (II) to create this catalog. In the following examples, the VSAM catalog named MQMCAT is used, and it is assumed that its label is already defined in the disk label area.

b. Allocate a VSE library.

This step is not required if you restore the product into the PRD2 library. However, if you want to install MQSeries in another library, you must create one. You are recommended to use the Interactive Interface dialogs for creating this library, or run the following sample adapted for your environment.

If you adapt this sample you must modify all the other provided samples accordingly, by changing the following fields:

- NAME from VSE.MQMUSR1.LIBRARY to your selected name
- VOLUMES from vol id to your volume identifier
- DATA (NAME from VSE.MQMUSR1.LIBRARY.DATA to your selected data name
- CATALOG from cata log.name to your catalog name

```

* $$ JOB JNM=DEFLIB,CLASS=0,DISP=D
// JOB   DEFINE MQSeries Library MQMUSR1
// EXEC IDCAMS,SIZE=AUTO
DEFINE CLUSTER (
    NAME (VSE.MQMUSR1.LIBRARY)
    CYLS (3 1)
    SHAREOPTIONS (3)
    RECORDFORMAT (NOCIFORMAT)
    VOLUMES (volid)
    NOREUSE
    NONINDEXED
    TO (99366))
DATA (NAME (VSE.MQMUSR1.LIBRARY.DATA))
CATALOG (catalog.name)
/*
/ &
* $$ E0J

```

2. Restore the MQSeries sublibrary from the library tape. You can do this by either:

a. Using the Interactive Interface Dialogs, as follows:

- 1) From an administrator ICCF signon, select the “Installation” option.
- 2) Select “Install Programs – V2 format”.
- 3) Select “Prepare for installation”.

This presents you with a series of panels and options to identify the tape address and process a job, by scanning the mounted tape and identifying which stacked products are available for installation.

Monitor the VSE console to see when this job has completed. When it has completed, proceed to the step 2a4.

- 4) Select “Install Program(s) from Tape”.

You are presented with a list of products available from the install tape and suggested install sublibraries. You can select either the default install library, “PRD2.MQSERIES”, or the name of the customized library you created in Step 1 on page 10.

- 5) Select option 1 to proceed with the installation and press function key five (PF5) to create a job to be submitted.

or

b. Customizing and processing the following JCL, using the library name from step 1 on page 10.

```

* $$ JOB JNM=MQMTAPE,CLASS=0,DISP=D
// JOB MQMTAPE Restore MQSeries from tape
// ASSGN SYS006,cuu
// MTC REW,SYS006
// EXEC MSHP,SIZE=1M
INSTALL PRODUCT FROMTAPE ID='MQSeries..2.1.0' -
PROD INTO=lib.sublib
/*
/ &
* $$ E0J

```

New user procedures

Where:

- cuu** Is the tape drive address
- lib.sublib** Is the sublibrary into which the product is to be installed, for example, PRD2.MQSERIES

Installation checkpoint (MQSeries installation)

You should now have correctly installed the MQSeries sublibrary. This can be verified using a VSE Librarian job to inspect the contents of the library.

The MQSeries phases, objects, and sample jobs are visible.

Note: If the MQSeries product has not installed correctly, check through the preceding instructions to ensure that they all completed correctly.

If you are a new user, see “Procedures for new users.” If you are migrating to MQSeries for VSE/ESA V2.1 from an earlier release, see “Migration procedures for existing users” on page 24.

Procedures for new users

The following steps describe how to

- Allocate and initialize the required MQSeries files
- Customize your CICS system to utilize the MQSeries facilities

The samples for the following jobs can be found in the installation library you selected, or “PRD2.MQSERIES”.

Allocate and initialize the required MQSeries files

You must now run the jobs to:

- Create the setup file
- Create the MQSeries configuration file
- Create cluster definitions for MQSeries queues

Note to users

The sample JCL jobs **must** be modified and customized to refer to your own volume identifiers and catalog names.

This should be done by your VSE systems programmer.

MQJSETUP.Z

Allocate a VSAM ESDS, MQFSSET, which is needed to populate the MQSeries configuration file with text and help messages at initialization time.

MQJCONFIG.Z

Allocates the MQSeries (CICS) subsystem configuration file. For this VSAM KSDS file, each record is a fixed length of approximately 2 KB.

To estimate the space you require, allocate one record, consisting of one cylinder for normal operation, for each MQSeries channel and queue.

MQJQUEUE.Z

Allocates and initializes the MQSeries message queue files. For these VSAM KSDS files, each record is of varying length, depending upon the size of the user data area. A message queue file is required for each queue defined to the MQSeries (CICS) subsystem.

To estimate the space required for each message queue, use the following guidelines:

- Each message queue file contains one header record for each local queue.
- One record is written for each user message.
- Each record is of variable length and consists of a header of 740 bytes plus the actual variable-length user data area.
- This job allocates the following message queue files:

MQSERIES.MQFERR – Dead letter queue file
 MQSERIES.MQFLOG – Error log queue file
 MQSERIES.MQFMON – Monitor queue file

The following files are sample definitions for user message queues:

MQSERIES.MQFI001
 MQSERIES.MQFO001
 MQSERIES.MQFI002
 MQSERIES.MQFO002
 MQSERIES.MQFI003
 MQSERIES.MQFO003

You are strongly recommended to define one local queue in each physical file.

Preparing CICS for MQSeries

Various CICS tables and definitions must be created and customized for use by the MQSeries subsystem.

You must define the following:

- CICS resources into the CICS CSD
- Entry definitions for the CICS Destination Control Table
- Entry definitions for the CICS File Control Table

The definitions should be reviewed by your CICS systems programmer.

Use the samples (see Appendix E, “Sample programs” on page 145) provided with the product. See Appendix B, “CICS control table definitions” on page 113 for further information.

To help you install the PCT and PPT CICS definitions, the sample MQJCSD.Z is provided. MQJCSD.Z automatically defines the MQSeries entries required into the CICS Definition Data Set (without using migrated CICS, DFHPPT and DFHPCT tables).

You may need to modify this sample to fit your own environment, because all entries are defined in group “MQM”, which is then added to the VSELIST list.

MQJCSD.Z – Define CICS resources into the CICS CSD

Sample code that can be used to create CICS-specific PCT and PPT definitions, which are required by the MQSeries subsystem.

MQCICFCT.Z – File Control Table (FCT)

The sample code provided can be used for creating CICS definitions for the MQSeries configuration and sample queue files. These definitions may require changing to your site's specific requirements.

MQCICDCT.Z – Destination Control Table (DCT)

The MQSeries product requires intrapartition transient data queues (TDQ), CSMT and MQER, for the processing of log messages. This sample provides a suitable definition for MQER.

Modify CICS start-up deck

For CICS applications to use the MQSeries facilities, you must inform CICS of the MQSeries configuration and work files, and the location of the MQSeries for VSE/ESA phases as follows:

- Add the label definitions for the CICS start-up job (MQJLABEL.Z) to your CICS start-up deck, or to the standard label procedures. It contains information about the datasets that MQSeries for VSE/ESA uses.

Note to users

This file **must** be modified and customized to refer to the correct volume identifiers and catalog names.

This should be done by your VSE systems programmer.

- Add the MQSeries for VSE/ESA subsystem install library defined in "Installing MQSeries for VSE/ESA – all users" on page 10 (default name "PRD2.MQSERIES") to the LIBDEF control statement in your CICS startup deck.

For example:

```
// LIBDEF          *, SEARCH (PRD2.MQSERIES,      *
                  PRD2.CONFIG,                  *
                  PRD2.SCEECICS,                 *
                  PRD2.SCEEBASE,                 *
                  PRD2.CICSR)
```

Recovery and restart

Although MQSeries uses its own recovery and restart logic, it also uses standard CICS file management. It is important, therefore, that you define all the MQSeries VSAM clusters in the DFHFCT with the LOG parameter set to YES. In addition, ensure that the CICS logging facility is activated with JCT = xx or YES in your DFHSIT.

If you do not fulfil the above conditions, unpredictable results can occur. For example, loss of messages or inaccurate values for message sequence numbers may occur.

CICS journal control table

The CICS journal control table (JCT) can be affected by the queue definitions. If a physical record is larger than the buffer size specified in the JCT, a CICS task abend of “AFCL” occurs.

The provided sample FCT queue definitions specify a maximum record length of 4089 bytes. If large records are written, you should set the BUFSIZE parameter of the CICS DFHJCT to a different value; a BUFSIZE value of 4200 is usually sufficient.

For further information, see the *CICS/VSE Resource Definition (Macro)* manual.

This is reflected in either the MQSeries System Log or the CSMT TD queue when an MQPUT call is processed trying to perform this function.

Uppercase translation

Queue manager, queue and channel names are case sensitive on MQSeries systems. If MQSeries on VSE/ESA sends messages to other MQSeries systems, you must specify UCTRAN = TRANID or UCTRAN = NO in your CICS terminal definitions.

If you do not do this, the names you enter into the MQSeries panels are translated into uppercase, and they may not match the actual names on the target MQSeries system.

Installation checkpoint (CICS)

You have now set up the CICS system, and it is ready to be restarted to update the system configuration and utilize the MQSeries subsystem.

Note: If the CICS system has not been updated correctly, check through the preceding instructions to ensure that they all completed correctly.

Starting MQSeries

When you have restarted the CICS system, you need to initialize and populate the MQSeries for VSE/ESA configuration file before you can use the MQSeries subsystem.

You do this with the MQSU transaction. However, you are strongly recommended to ensure that all the MQSeries for VSE/ESA subsystem files are available for access by CICS before running this job.

You do this by issuing the CICS transaction:

```
CEMT INQUIRE FILE(MQF*)
```

All of the MQSeries for VSE/ESA files defined in “Allocate and initialize the required MQSeries files” on page 12 should be visible, and you should be able to open, close, enable, and disable the files.

If you cannot access these files, refer to your CICS systems programmer and review the steps in “Preparing CICS for MQSeries” on page 13.

Starting MQSeries

If the files are accessible, issue the transaction MQSU. This completes with the message “MQSU – MQSeries Install Completed”.

Note that you need only run the MQSU transaction once whenever you install MQSeries for VSE/ESA. This rule applies to initial installations of the product, as well as subsequent installations.

MQSeries initialization

Initialize your new MQSeries for VSE/ESA system as follows:

1. Set up the MQSeries for VSE/ESA environment.

Issue MQSE (Setup Environment).

The response “MQSE:MQSeries environment setup completed” is displayed, after approximately one minute.

2. Specify the queue manager name.

There can be only one queue manager on each MQSeries for VSE/ESA system and each MQSeries system should have a unique queue manager name. The name is specified using the MQMT System Administration transaction, as follows:

- a. Enter the transaction code MQMT on a CICS terminal.
- b. Select option 1 for the “Configuration” menu.
- c. Select option 1 for the “Global System Definition” update screen.

```
07/28/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
18:28:25        Global System Definition                VSE1
MQMMSYS         Queue Manager Information              A001
Queue Manager . . . . . : VSE.QMANAGER
Description Line 1. . . . : MQSeries for VSE/ESA V2
Description Line 2. . . . :
                    Queue System Values
Maximum Number of Tasks . . : 00000100      System Wait Interval : 00000005
Maximum Concurrent Queues . : 00000100      Max. Recovery Tasks  : 0000
Allow TDQ Write on Errors  : Y   CSMT       Allow Internal Dump   : Y
                    Queue Maximum Values
Maximum Q Depth . . . . . : 01000000      Maximum Global Locks.: 00000100
Maximum Message Size. . . . : 00002048      Maximum Local Locks .: 00000100
Maximum Single Q Access . . : 00000100      Checkpoint Threshold : 1000
                    Global QUEUE /File Names
Local Code Page . . . : 00500      TCP/IP Listener Port : 00000
Configuration File. . : MQFCNFG      Licensed Clients . . : 00000
LOG Queue Name. . . : SYSTEM.LOG
Dead Letter Name. . . : SYSTEM.DEAD.LETTER.QUEUE
Monitor Queue Name. . : SYSTEM.MONITOR

Record initialized - New record.
PF2 = Main Config  PF3 = Quit      PF4/ENTER = Read      PF6 = Update
```

Figure 1. Default screen

- d. Change the “Queue Manager” field to the name that you are giving to your local queue manager.
- e. Press function key six (PF6) to update the configuration.
- f. Press function key three (PF3) to quit the screen.

You can leave the other fields unchanged.

3. Define a local queue.

You must define some local queues to test the operation of the MQSeries for VSE/ESA subsystem. This task is also carried out by using the MQMT transaction.

The following definitions allow the installation verification program, TST2, to send messages to ANYQ.

Carry out the following procedure:

- a. Type MQMT at the system prompt.
 - b. Type 1 on the main menu to select Configuration.
 - c. Type 2 on the Configuration menu to select queue definitions. The "Queue Main Options" screen appears.
 - d. Complete the following fields:
 - Object Type L
 - Object Name ANYQ
 - e. Press PF5 (Add) to display the "Local Queue Definition" screen.
 - f. Press PF5 (Add) to display the "Queue Extended Definition" screen and change the default values in the following fields:
 - Usage mode N (Normal)
 - Physical File Name MQFI001 (file name from FCT)
 - Maximum Q Depth 00000100
 - Maximum Message Length 00002048
 - g. Press PF5 (Add) to save the changes.
 - h. Press PF2 (Options) to return to the Queue Main Options Screen.
 - i. Press PF9 (List) to display a selection screen.
 - j. On the selection screen, use the cursor keys to select the queue. Press any character key followed by the Enter key.
- A screen displays the queue parameters that you have entered. Check that the correct data has been entered.

4. Initialize the MQSeries for VSE/ESA queue manager. There are two ways of doing this. Either:

- a. Type MQIT on a CICS terminal.

The response "MQIT: No channel definitions. Initialization completed" is displayed when the process has completed. This is normal.

Attention: The first time you issue the command MQIT, the following warning message appears if you attempt to initialize the system before you define a queue manager name.

```
MQIT: No system record has been defined
```

This is normal, and disappears once the system configuration record has been defined in MQMT.

or

- b. Use the MQSeries for VSE/ESA System Administration transaction (MQMT), as follows:

Starting MQSeries

- 1) Issue MQMT to display the main menu panel of MQSeries Administration.
- 2) Select 2 - Operation.
- 3) Select 4 - Initialization/Shutdown.
- 4) Type I in the function field and press function key six (PF6).

Note: If you carry out the initialization before you perform system setup, you receive the message MQ900000:MQSERIES VSE ENVIRONMENT NOT INITIALIZED.

In the future, you can combine Step 1 on page 16 and Step 4 on page 17 by issuing MQSE with the parameter I to perform the initialization step, as follows:

```
MQSE I
```

The response “MQSE:MQSeries environment setup and initialized” is displayed when the process has completed.

Defining the SYSTEM.LOG queue

When you have completed the steps listed in “MQSeries initialization” on page 16, the queue manager is active and you can verify this by typing MQMT on a CICS console, to display Figure 2.

```
01/28/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYZMZSI2
08:39:08      *** Configuration Main Menu ***          VSE2
MQMMCFG                                               0002

                SYSTEM HAS BEEN SHUTDOWN

                Maintenance Options :
                  1. Global System Definition
                  2. Queue Definitions
                  3. Channel Definitions

                Display Options      :
                  4. Global System Definition
                  5. Queue Definitions
                  6. Channel Definitions

                Option:

Please enter one of the options listed.
5786-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process      PF2 = Main Menu      PF3 = Quit
```

Figure 2. Configuration main menu

Ensure that the “SYSTEM HAS BEEN SHUTDOWN” message is displayed.

The MQSeries for VSE/ESA subsystem records log messages to a standard MQSeries queue with the name “SYSTEM.LOG”. This queue needs to be defined to the system, as with any other queue.

To do this, follow the instructions listed in Step 3 on page 17 and set the:

- Queue name to “SYSTEM.LOG”
- Physical file name to “MQFLOG”

MQSeries installation verification test

The MQSeries subsystem is now ready for the installation verification procedures.

Stop the MQSeries subsystem, using either the MQST transaction, or the Operations Shutdown menu – MQMT option 2.4, and then reinitialize the MQSeries subsystem (see “MQSeries initialization” on page 16).

To carry out the installation verification test you need:

- One local queue
- The sample transaction TST2
- The program TPTTST2 provided with the product
- Access to two terminals

Local queue verification test

The local queue verification test consists of five steps:

1. Initialize the MQSeries run time environment.
2. Use the test program TPTTST2 to send a number of messages.
3. Use MQMT to verify that these messages are on the queue.
4. Use the test program TPTTST2 to read the messages.
5. Use MQMT to verify that the messages have been delivered.

Step 1 (initializing the MQSeries run time environment) has already been performed. You only need to test the ability to send and receive messages, as follows:

1. On one terminal issue the transaction code TST2. This invokes the MQSeries for VSE/ESA test program TPTTST2 and produces the following screen:

```

TST2 is a test facility for SENDING / RECEIVING messages
The format of command is as follows:
TST2 XXXX [NN] QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ

NOTE a single space or comma separates the parameters)
XXXX 4-character function code, pad with trailing blank
      HELP - DISPLAY THIS HELP TEXT
      PUT  - MQPUT  MESSAGES
      PUT1 - MQPUT1 MESSAGES
      PUTR - MQPUT W/ REPLY MESSAGE
      GET  - MQGET  MESSAGES
      GETD - MQGET W/ BROWSE & DELETE
      BOTH - MQPUT FOLLOWED BY MQGET
      INQ  - INQ ABOUT QUEUE (no count NN)
NN    2-digit number with leading zero (01 TO 99)
QQQQ  A 48-character field giving the name of a queue.
An additional prompt will ask for the name of the reply queue for PUTR option.

```

Figure 3. TPTTST2 screen

2. On a second terminal, start MQMT and use option 3.1 to monitor queue operations. This displays the following screen:

Installation verification test

```
07/06/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
15:06:30           Monitor Queues                               VSE1
MQMMMOQ                                                    A001

                    QUEUING SYSTEM IS ACTIVE

QUEUE              FILE      T INBOUND  OUTBOUND  LR      QDepth
ANY                MQFI001 N IDLE    IDLE      0       0
SYSTEM.LOG        MQFO001 N IDLE    IDLE      0       1

Information displayed.
  ENTER = Refresh      PF2 = Main Monitor
  PF7 = Back   PF8 = Forward   PF9 = All   PF10 = Detail
```

Figure 4. Monitor queues screen

3. On the first terminal, issue:

```
TST2 PUT 10 ANYQ
```

Note: If you type TST2 without parameters, the HELP screen for using TPTST2 is displayed.

4. TPTST2 sends the specified messages addressed to ANYQ.

You receive the following message on successful completion of the transaction:

```
FULL CYCLE HAS BEEN PERFORMED SUCCESSFULLY
QUEUE USED - ANYQ
NUMBER OF MESSAGES PROCESSED - 10
TOTAL SECONDS ..... - hh:mm:ss
```

where:

- 10 is the number of messages you specified (nn).
- hh:mm:ss is the time taken to process nn messages.

5. Return to the terminal running the MQMT Monitor Queue process.

6. Press the Enter key on this terminal.

The QDEPTH column for queue ANYQ now equals 10. This is the value you specified for nn Step 4.

7. Messages on an MQSeries for VSE/ESA queue can be displayed at any time using the MQMT Browse Queue facility (MQMT option 4). Select this option, enter the queue name in the "Object" field, and press the Enter key.

This displays the following:


```

07/06/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
15:37:45      Browse Queue Records      VSE1
MQMDISP      SYSTEM IS ACTIVE      A001

      Object Name: ANYQ
      QSN Number : 00000001      LR-      0, LW-      10, DD-MQFI001
      Queue Data Record
Record Status : Written.      PUT date/time : 19980706153733
Message Size : 00000200      GET date/time :
Queue line.
THIS IS A MESSAGE TEXT

Information displayed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process PF2 = Main Menu PF3 = Quit PF4 = Next PF5 = Prior
PF7 = Up PF8 = Down PF9=Hex/Char PF10=Txt/Head
    
```

Figure 5. Browse Queue Records screen – status written

The queue can then be browsed forwards and backwards using function keys four and five (PF4 and PF5).

Note that in this example, the “Record Status” field is *Written*. This indicates that the message has been placed on the queue but not retrieved.

8. Move to the other terminal.
9. At the CICS prompt, type:

```
TST2 GETD 10 ANYQ
```

Note: If you type TST2 without parameters, the HELP screen for using TTPTST2 is displayed.

10. TTPTST2 reads the specified messages from ANYQ.

You receive the following message on successful completion of the transaction:

```

FULL CYCLE HAS BEEN PERFORMED SUCCESSFULLY
QUEUE USED - ANYQ
NUMBER OF MESSAGES PROCESSED - 10
TOTAL SECONDS ..... - hh:mm:ss
    
```

where:

- 10 is the number of messages you specified (nn).
- hh:mm:ss is the time taken to process nn messages.

11. Return to the terminal, running the MQMT Monitor Queue process.
12. Press the Enter key. The Monitor Queue screen still displays ANYQ as the only defined queue.

Notes:

- a. The QDEPTH number, representing the number of messages on the queue, has decreased to zero.
 - b. The total number of messages read from the queue (LR) has increased by the number you read using TTPTST2.
13. Use the MQMT Browse facility to view ANYQ. The “Record Status” field has changed to Deleted, and the “GET date/time” field is now completed.

This indicates that the record has now been retrieved by an application. In this case the test transaction TST2 was used with parameters “TST2 GETD 5 ANYQ”.

```
07/06/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
15:39:13            Browse Queue Records                          VSE1
MQMDISP            SYSTEM IS ACTIVE                               A001

Object Name: ANYQ
QSN Number : 00000001      LR-      5, LW-      10, DD-MQFI001
Queue Data Record
Record Status : Deleted    PUT date/time : 19980706153733
Message Size : 00000200   GET date/time : 19980706153910
Queue line.
THIS IS A MESSAGE TEXT

Information displayed.
5686-A06 (C) Copyright IBM Corp. 1998          All Rights Reserved.
ENTER = Process  PF2 = Main Menu  PF3 = Quit  PF4 = Next  PF5 = Prior
PF7 = Up        PF8 = Down      PF9=Hex/Char  PF10=Txt/Head
```

Figure 6. Browse Queue Records screen – status deleted

Note that MQSeries for VSE/ESA differs from many other MQSeries platforms, in that when a message is retrieved from a queue it is logically deleted but not physically deleted. The messages are merely flagged as “Deleted”.

As a consequence of this technique of flagging messages as “written” and “deleted”, messages can have their logical state changed, and where necessary, reprocessed.

You can do this using MQMT option 2.5. However, you are advised to carry out this procedure only when you are familiar with the MQSeries for VSE/ESA system.

You have now completed a local installation verification test demonstrating that two applications can send and receive messages through an MQSeries queue.

Remote queue verification test

In order to expand this test to include a remote link, you must carry out the following steps:

1. Using the appropriate manufacturer's directions, install the prerequisite hardware and software required to support the selected transport protocol (SNA LU6.2 or TCP/IP).
2. Define the MQSeries channels that you require. See "Channel definitions" on page 60, and coordinate this task with the remote system administrator.
3. Configure the transmission queues and remote queues required by MQSeries to communicate over the channel – see "Channel definitions" on page 60.

In order for new queue definitions and channels to take effect at run time, you must shut down MQSeries by:

1. Closing the channels.
2. Stopping the queues.
3. Shutting down MQSeries for VSE/ESA.

You must then reinitialize MQSeries as described in "MQSeries initialization" on page 16.

You have now installed and locally verified MQSeries and you can use the administrative programs and the MQI libraries.

However, before your user applications can effectively use the system for message transmission, you must fully configure the system with your queue definitions.

This last step is the most important part of the installation. The requirements are detailed in:

- Chapter 3, "Configuring network communications" on page 29, which provides the configuration guidelines.
- Chapter 4, "System operation" on page 41, which describes the MQSeries System administration screens used in the configuration.

Installation checkpoint (installation verification test)

You can now:

- Define local queues.
- Start and stop the queue manager.
- Browse queues using MQMT.
- Monitor the status of queues.
- Run simple MQSeries programs that use local queues.

Note: If the installation verification test has not completed, check through the preceding instructions.

Post installation verification test CICS modifications

The MQSeries for VSE/ESA subsystem can be started and stopped automatically as part of the normal CICS start up and shutdown procedures. You do this by adding appropriate entries to the CICS Initialization and Shutdown parameters.

Note to users

You **must** not carry out these steps until you have installed MQSeries for VSE/ESA.

CICS Program List Table Post Initialization (PLTPI)

The MQSeries subsystem requires initialization before applications can start using the queue manager. These steps setup the MQSeries environment and initialize the MQSeries resources.

To start MQSeries automatically, you can add the following programs to the CICS initialization PLT (PLTPI) list:

MQPSENV Set up the MQSeries environment.
MQPSTART Initialize the resources.

For example:

```
DFHPLT TYPE=ENTRY, PROGRAM=MQPSENV  
DFHPLT TYPE=ENTRY, PROGRAM=MQPSTART
```

Other methods are given in “MQSeries initialization” on page 16.

CICS Program List Table Shut Down (PLTSD)

The MQSeries subsystem should be shutdown correctly before shutting down CICS. This can be done:

- Manually, using transaction MQST
- Automatically, by placing the MQSeries program MQPSTOP in the CICS shutdown PLT before the DFHDELIM statement

For example:

```
DFHPLT TYPE=ENTRY, PROGRAM=MQPSTOP
```

This ensures that MQSeries ends during the first phase of CICS shutdown.

Migration procedures for existing users

You are strongly recommended to review the sections “Installing MQSeries for VSE/ESA – all users” on page 10 and “Procedures for new users” on page 12 before proceeding with this section.

You should pay particular attention to those sections concerned with CICS definitions, because some new PPT and PCT definitions have been created for this release. You must ensure that these definitions have been added to your CICS system.

The structure and layout of the MQSeries for VSE/ESA configuration and queue files has changed between Version 1.4 and Version 2.1. This will require the deletion and recreation of these files.

The definitions in the configuration file can be migrated using the supplied utilities; however, the data in the queue files must be processed.

The following steps are required for successful migration from Version 1.4 to Version 2.1:

1. Ensure that all local and transmission queues are fully processed, with depth equal to zero, and that any required data is securely backed up.
2. Stop the MQSeries for VSE/ESA subsystem.
3. Backup your old MQSeries for VSE/ESA configuration file, MQFCNFG.
4. Offload your current MQSeries for VSE/ESA definitions using the MQJMIGR1.Z supplied sample program, to produce MQSERIES.MQOCNFG.

The supplied sample code must be reviewed, and customized to reflect your system specific volume and catalog names.

```

* ** JOB JNM=MQJMIGR1,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJMIGR1 - Migrate MQ/Series for VSE/ESA Configuration file.
* -----*
*   I M P O R T A N T   I M P O R T A N T   I M P O R T A N T   *
*
*   Please change :
*
*       "* ** JOB" to "* $$ JOB"
*       "* ** LST" to "* $$ LST"
*       "* ** EOJ" to "* $$ EOJ"
*
*   Fields marked with ?vol-id?, ?cat-id? or ?cat-name? must be
*   modified to suit user specifications.
*
* -----*
*
*   Use this sample only to migrate from Version 1.4 to Version 2.1
*
*   This job extracts your system, queue and channel definitions
*   from a Version 1.4 configuration file.
*   It then reformats them into the new Version 2.1 format.
*   New formatted records are added to a work file (MQOCNFG) to be
*   merged later into the new configuration file defined by the job
*   MQJCONFG and initiated by the CICS transaction MQSU.
*   The merge process may then be executed (see job MQJMIGR2).
*
* -----*
* Licensed Materials - Property of IBM
*
* 5686-A06
* (C) Copyright IBM Corp. 1998
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
* -----*
*
// DLBL OLDCNFG,'MQSERIES.MQFCNFG',,VSAM,CAT=?cat-name?
// DLBL NEWCNFG,'MQSERIES.MQOCNFG',,VSAM,CAT=?cat-name?
// EXEC IDCAMS,SIZE=AUTO
/*                                     */
/*   VERIFY VSAM FILE, CANCEL THE JOB IF IT IS IN USE   */
/*
/*       VERIFY FILE(OLDCNFG)
/*       IF MAXCC > 0 THEN CANCEL
/*
/*                                     */
/*                                     */
/*   DELETE AND DEFINE THE WORK FILE
/*
/*                                     */
/*                                     */
/*   DELETE   (MQSERIES.MQOCNFG)
/*           CL ERASE PURGE CAT(?cat-id?)
/*   SET MAXCC = 0

```

Migration procedures

```
DEFINE CLUSTER                                -
      (NAME (MQSERIES.MQOCNFG)                -
      RECORDS (50 10)                         -
      RECORDSIZE (2048 2048)                  -
      VOLUMES (?vol-id?)                      -
      KEYS (100 0)                            -
      SHR (2)                                  -
      INDEXED)                                -
      DATA                                    -
      (NAME (MQSERIES.MQOCNFG.DATA) CISZ(4096)) -
      INDEX                                    -
      (NAME (MQSERIES.MQOCNFG.INDEX) CISZ(512)) -
      CAT (?cat-id?))
/*
// IF $MRC > 0 THEN
// GOTO NOPROC
// LIBDEF PHASE,SEARCH=(PRD2.MQSERIES,PRD2.SCEEBASE)
// EXEC MQPCONFG,SIZE=AUTO
/*
/. NOPROC
/&
* ** EOJ
```

5. Create the following files using the supplied sample program shown:
 - a. A new message file using MQJSETUP.Z.
 - b. A new configuration file using MQJCONFIG.Z.
 - c. New queue files using MQJQUEUE.Z.

Note: The files MQJSETUP.Z, MQJCONFIG.Z, and MQJQUEUE.Z, which are described in “Allocate and initialize the required MQSeries files” on page 12, need to be customized to reference your site-specific names.

6. Restart CICS, ensuring that the new CICS definitions, and the new phases installed in the sublibrary described in “Installing MQSeries for VSE/ESA – all users” on page 10. have been included.

The default sublibrary name is “PRD2.MQSERIES”. If you normally use PLTPI to start MQSeries for VSE/ESA, ensure that PLTPI does **not** run during this initial start up.

7. Run the MQSU transaction to populate the configuration file with new message definitions

This transaction completes by displaying the message “MQSeries Install completed”.

8. Migrate the configuration definitions by using the following procedure:

- a. Use the CEMT transaction to close the configuration file, by issuing the command:

```
CEMT SET FILE(MQFCNFG) CLOSED
```

- b. Customize the sample MQJMIGR2.Z to refer to your site-specific names, and migrate the old definitions from MQSERIES.MQOCONFG using the customized version of MQJMIGR2.Z.
- c. Process the job MQJMIGR2.Z.

```

* ** JOB JNM=MQJMIGR2,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJMIGR2 - Migrate MQ/Series for VSE/ESA Configuration file.
*-----*
*           I M P O R T A N T           I M P O R T A N T           I M P O R T A N T           *
*                                                                                                     *
*   Please change :                                                                                       *
*           "* ** JOB" to  "* $$ JOB"                                                                 *
*           "* ** LST" to  "* $$ LST"                                                                 *
*           "* ** EOJ" to  "* $$ EOJ"                                                                 *
*                                                                                                     *
*   Fields marked with ?vol-id?, ?cat-id? or ?cat-name? must be *
*   modified to suit the user specifications.                                                         *
*-----*
*                                                                                                     *
*   Use this sample only to migrate from version 1.4 to 2.1.                                         *
*                                                                                                     *
*   This job has to be executed only if the previous migration steps *
*   have been successfully processed. That is :                                                       *
*                                                                                                     *
*   - Reformat old configuration file to a work file (job MQJMIGR1) *
*   - Redefine a new configuration file (job MQJCONFG)                                                *
*   - Fill up the new configuration file (transaction MQSU)                                           *
*                                                                                                     *
*   Thus job merges records saved to the work file into the new *
*   configuration file, then deletes the work file                                                    *
*-----*
* Licensed Materials - Property of IBM                                                                 *
*                                                                                                     *
* 5686-A06                                                                                             *
* (C) Copyright IBM Corp. 1998                                                                       *
*                                                                                                     *
* US Government Users Restricted Rights - Use, duplication or *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
*-----*
// DLBL MQFCNFG,'MQSERIES.MQFCNFG',,VSAM,CAT=?cat-name?
// DLBL MQOCNFG,'MQSERIES.MQOCNFG',,VSAM,CAT=?cat-name?
// EXEC IDCAMS,SIZE=AUTO
/*                                                                                                     */
/*   VERIFY VSAM FILES, CANCEL THE JOB IF THEY ARE IN USE                                           */
/*                                                                                                     */
/*           VERIFY FILE(MQFCNFG)                                                                     */
/*           VERIFY FILE(MQOCNFG)                                                                     */
/*           IF MAXCC > 0 THEN CANCEL                                                                 */
/*           REPRO INFILE(MQOCNFG) OUTFILE(MQFCNFG) REPLACE                                          */
/*                                                                                                     */
/*           DON'T ERASE THE WORK FILE IF REPRO FAILED                                               */
/*                                                                                                     */
/*           IF MAXCC > 0 THEN CANCEL                                                                 */
/*           DELETE (MQSERIES.MQOCNFG) CL NOERASE PURGE -                                           */
/*             CATALOG(?cat-id?)                                                                     */
/*                                                                                                     */
/&
* ** EOJ

```

This job merges your old definitions into the new configuration file.

- d. Use the CEMT transaction to open the configuration file, by issuing the command:

```
CEMT SET FILE(MQFCNFG) OPEN
```

9. Use the MQMT transaction to verify that the global system queue and channel definitions are correct before you delete the old configuration file.

You can now restart MQSeries for VSE/ESA and run the installation test procedures. You can use your normal procedures to perform this, or issue the command MQSE I to run the environment setup and initialization in a single step.

Migration procedures

You are recommended to perform the installation verification procedures described in “MQSeries installation verification test” on page 19.

Note: Although existing Version 1.4 applications (VSE PHASES) are compatible with Version 2.1, you are advised to recompile all your applications in order to take advantage of the new API object code. This code has improved error and condition handling.

Installation checkpoint (migration)

You have now successfully migrated your Version 1.4 definitions to Version 2.1. You can view these definitions using the MQMT transaction. Your old queue files have been deleted and recreated.

Note: If the migration has not completed correctly, check through the preceding instructions.

Chapter 3. Configuring network communications

This chapter describes the steps you perform to configure MQSeries to run on the CICS system and communicate with other MQSeries systems. The chapter assumes that your chosen communications software has been installed and correctly configured on your system.

For ACF/VTAM®, using MQSeries should not require any changes to the:

- VTAM start parameters
- Definition of CICS systems to VTAM

However, you must define all the LUs that are involved.

For TCP/IP, using MQSeries with the TCP/IP communications protocol requires the installation of TCP/IP for VSE/ESA V1.3 or later.

TCP/IP is shipped as part of the VSE/ESA base product in library PRD1.BASE, and simply requires that you install a product key together with your customer information. For further details refer to the *TCP/IP for VSE/ESA User's Guide*.

MQSeries for VSE/ESA does not have any special TCP/IP installation or configuration requirements.

Note: If TCP/IP is to be used as a transport protocol, the TCP/IP phase library must be added to the LIBDEF statement in the CICS start up JCL before the SCEEBASE library.

This is because SCEEBASE contains a TCP/IP phase stub that handles TCP/IP API calls when TCP/IP is not installed.

This chapter describes how to configure:

- The queue manager
- Queues
- Channels

MQSeries system definitions required

The local MQSeries for VSE/ESA system has to be informed about remote MQSeries systems with which it will communicate. MQSeries has to be defined to:

- MQSeries on CICS (in the network specific parts of the channel definition)
- CICS itself, in one of the following ways:
 - In a TERMINAL definition
 - In CONNECTION/SESSION definitions
 - Through the CICS AUTOINSTALL facility
- VTAM (either predefined, or by VTAM dynamic resource definition), if you are using SNA LU6.2.

Definitions in CICS

If the CICS end of an MQSeries channel is to initiate the channel connection (that is, the CICS channel-endpoint is a sender), CICS performs an EXEC CICS ALLOCATE. However, this succeeds only if CICS is:

- A contention winner
- Already bound
- Not already allocated

If CICS has no definition of the resource, CICS is incapable of formulating a request to VTAM for session establishment. In these circumstances, CICS AUTOINSTALL is inappropriate – autoinstall is for incoming session establishment requests, not for outgoing ones.

Therefore, for sender channel-endpoints on VSE, a definition of the remote system is required at the CICS level.

If the remote system, at the network level, is capable of supporting parallel sessions (for example, it has independent LU6.2 capability, or it is another CICS system) and, you intend to configure several channels between the two systems, you should use CONNECTION and SESSIONS definitions.

Typical definitions, using the CICS Resource Definition Online (RDO) transaction, CEDA, are shown in Figure 7.

```
DEFINE GROUP(<group name 1>)
CONNECTION(<remote conn>)
NETNAME(<remote luname>)
ACCESSMETHOD(VTAM)
PROTOCOL(APPC)
SINGLESESS(NO)

DEFINE GROUP(<group name 1>)
SESSIONS(<sess name>)
CONN(<remote conn>)
MODE(<logmode 1>)
MAXIMUM(<max sessions>,<max CICS contention winners>)

INSTALL GROUP(<group name 1>)

ADD GROUP(<group name 1>) LIST(<start-up list>) {AFTER(<group name>)}
```

Figure 7. Definitions in CICS using RDO for parallel session partner LU

If the remote LU is capable of only one session, then it may be defined to CICS as either a single-session connection definition or as a terminal definition (Figure 9 on page 31).

```

DEFINE GROUP(<group name 2>)
CONNECTION(<remote conn>)
NETNAME(<remote luname>)
ACCESSMETHOD(VTAM)
PROTOCOL(APPC)
SINGLESESS(YES)

DEFINE GROUP(<group name 2>)
SESSIONS(<sess name>)
CONN(<remote conn>)
MODE(<logmode 2>)
MAXIMUM(1,1)

INSTALL GROUP(<group name 2>)

ADD GROUP(<group name 2>) LIST(<start-up list>) {AFTER(<group name>)}

```

Figure 8. Definitions in CICS for single-session capable partner LU.

```

DEFINE GROUP(<group name 3>)
TERMINAL(<remote conn>)
NETNAME(<remote luname>)
TYPETERM(DFHLU62T)
MODENAME(<logmode 2>)

INSTALL GROUP(<group name 3>)

ADD GROUP(<group name 3>) LIST(<start-up list>) {AFTER(<group name>)}

```

Figure 9. Definitions in CICS singles-session capable LU

The CICS supplied typeterm definition, DFHLU62T, provides a suitable terminal type definition. It exists in group DFHTYPE, which should be installed on your system.

Sample definitions for CICS tables can be found in the sublibrary PRD2.MQSERIES. However, other definitions are specific to your environment and you have to create them manually using the CEDA transaction, or DEFINE commands if using the DFHCSDUP batch program.

The definitions consist of a:

- Connection definition – see “Connection definition”
- Session definition – see “Session definition” on page 32

Connection definition

CICS uses the connection name to identify the other systems. For example, if sessions in VSE1 are to converse with sessions in VSE2 and MVS, you must define both VSE and MVS connections in each direction.

You must also define all the sessions and terminals involved if you are using SNA LU6.2.

Type CEDA DEF CONN GROUP(MQSERIES) to create connections, and set the fields to the following values:

MQSeries definitions

<i>Table 1. Object Characteristics of Connection</i>		
Category	Parameter	Desired Value
	Connection	VSE2
	Group	MQSERIES
Connection Identifiers	Netname	vse2lu62
Connection Properties	ACcessmethod	Vtam
	Protocol	Appc
	Datastream	User
	RECORDformat	U
Operational Properties	AUTOconnect	Yes
	INService	Yes
Security	ATTachsec	Local

The settings detailed, together with default values are sufficient for operation. For other parameters, refer to the *CICS/VSE 2.3 Resource Definition (Macro)* manual.

You can also display the connection status by typing CEMT I CEMT I CONN, to display:

<i>Table 2. CEMT I CONN display output.</i>	
STATUS: RESULTS - OVERTYPE TO MODIFY	
Conn(VSE2) Net(xxxxxxxx)	Ins Acq
Conn(MVS) Net(xxxxxxxx)	Ins Rel

Session definition

Type CEDA DEF SESSION G(MQSERIES) to create session names. Enter the values shown in the following table to complete the fields:

<i>Table 3 (Page 1 of 2). CEDA V SESS display parameter settings</i>		
Category	Parameter	Desired Value
	Sessions	VSE1VSE2
	Group	MQSERIES
Session Identifiers	Connection	VSE2
Session Properties	Protocol	Appc
	Maximum	00006,00003
	RECEIVEcount	No
	SENDCount	No
	SENDSize	04096
	RECEIVESize	04096

<i>Table 3 (Page 2 of 2). CEDA V SESS display parameter settings</i>		
Category	Parameter	Desired Value
Operational Properties	Autoconnect	Yes
	Buildchain	Yes
	RELreq	No
	Discreq	No
Recovery	RECOvoption	Sysdefault

The settings detailed, together with default values, are sufficient for operation. For other parameters, refer to the *CICS/VSE 2.3 Resource Definition Guide*.

Note: The DFHSIT Table must have the parameter ISC = YES to make the MQSeries system work.

MQSeries for VSE/ESA configuration guidelines

The following guidelines refer to the MQSeries master terminal (MQMT) administration dialogs. For information about using MQMT, see “MQSeries master terminal (MQMT) – main menu” on page 42.

There are three levels of configuration:

- The queue manager
- The channel
- The queue

Some fields are the same in all three levels, for example, the Maximum Message Size.

Notes:

1. The maximum message size defined in the queue manager configuration must be the largest of all those defined in the channels for this queue manager.
2. The size defined in the channel configuration must be equal to, or greater than, the largest message size that is accessing this channel.
3. Each level of maximum message size configuration utilizes different kinds of resources. Unnecessarily large sizes will consume address space.

Queue manager configuration guidelines

When configuring the queue manager (see “Global system definition” on page 44), use the following guidelines:

Maximum Number of Tasks

The maximum number (integer) of simultaneous connections to the queue manager. Though there is a slight overhead for each unused reservation, there is no harm in setting a large number, for example, 200.

Maximum Concurrent Queues

The maximum number of simultaneous open local queues allowed for the queue manager. You are recommended to set this to a large number, for example, 200.

System Wait Interval

The maximum polling time (in seconds) for the system monitor program after the system starts. A value of five seconds is usually sufficient.

Note: The system monitor task remains active until the CICS region is shut down, but exists in a wait state until the task is activated by the expiration of the System Wait Interval or by some specific application interface tasks.

The system monitor task starts up the trigger program and schedules the processes that reclaim resources held by applications that have ended abnormally. If there are too many, the System Wait Interval should be reduced to schedule this cleanup process more frequently.

Maximum Q Depth

The maximum number of active messages allowed by the queue manager for each queue. This value serves as the default Maximum Q Depth value when defining a queue. Any inbound message that causes the queue depth to exceed this size will be rejected as "Queue Full".

If this value is smaller than the Maximum Q Depth specified in the queue definition, it becomes the limiting value for the queue. You should set the value to double the maximum number of messages expected to be queued before any application starts to process them.

Maximum Message Size

The maximum number of characters allowed by the queue manager for each message. This field needs only to be large enough to accommodate the largest message. Setting a higher value than necessary wastes resource.

For example, if you anticipate the largest message to be 10 KB (10,240 bytes) you should set this field to 10240.

Note: Messages are stored in VSAM clusters and large messages can span multiple VSAM records. However, you should avoid spanning multiple clusters wherever possible, because of performance implications.

Where an entire message is stored within a single VSAM record, a message header of 740 bytes, for identification and description, is prefixed to the message.

Where a message is split across multiple records, each subsequent record uses a 56-byte header as a prefix to the data.

Maximum Single Q Access

This field defines the maximum number of MQOPEN calls against any queue handled by this queue manager. A value of 1000 calls is an acceptable value, if the maximum number of opens for each queue in the system is 100.

Maximum Global Locks

The maximum number of entries that the queue manager can use to maintain uncommitted MQPUT or MQGET calls, for each queue in the system, for recovery. A value of 500 is normally used.

Maximum Local Locks

The maximum number of entries that the queue manager can use to maintain uncommitted MQPUT or MQGET calls for each queue and task for recovery. Since an entry of a local lock is deleted once an application issues an explicit SYNCPOINT CICS command to commit updates, the more often an application takes the checkpoint, the fewer the maximum number of local locks needed.

You should specify a value greater than the largest message batch size for all the channel records. A value of 20 is usually sufficient.

Checkpoint Threshold

The maximum number of queue accesses between checkpoints to be taken by the queue manager. The smaller the value specified, the more often the queue manager takes checkpoints to secure the data integrity, thereby increasing the speed of recovery in the event of a system failure. The larger the checkpoint threshold, the fewer the resources that are used.

A value between 100 and 1000 is a good compromise between the requirements for performance and recovery speed. A high value can have data recovery implications.

Channel configuration guidelines

Defining the remote MQSeries system to the local queue manager is described in “Channel definitions” on page 60. However, from the point of view of showing where fields in the various definitions have to correspond, an outline MQSeries channel definition is shown in Figure 10.

```

07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
08:03:53           Channel Record          DISPLAY              MCHN
MQMMCHN           Last Check Point          Last Update 00000000    A005
MSN 00000002     Time 07:10:22   Interv 000000   Create Date 19980528
Name : SD01_TCP_VSEP
Protocol : L (L/T) Port : 1414   Type : R (S/R/C)
Partner :

      Allocation Retries          Get Retries
Number of Retries: 00000000      Number of Retries : 00000000
Delay Time - fast: 00000000      Delay Time         : 00000000
Delay Time - slow: 00000000

Max Messages per Batch : 000001      Max Transmission Size : 032000
Message Sequence Wrap  : 999999      Max Message Size     : 008192

Mess Seq Req(Y/N): Y   Convers Cap (Y/N): Y   Split Mssg(Y/N): N

Transmission Queue Name :
TP Name:
Checkpoint Values:      Frequency: 0000   Time Span: 0000
Enable(Y/N) Y          Dead Letter Store(Y/N) N
Channel record displayed.
PF2 =Menu PF3 =Quit PF4 =Read PF5 =Add PF6=Update PF9 =List PF12 =Delete

```

Figure 10. Outline MQSeries channel definition

Note that the TP Name field, which identifies the remote task identifier on the CICS system, need only be coded for a sender channel.

When configuring the channel, use the following guidelines:

Protocol

The required transport options for this channel. The options are:

- L – LU6.2 (SNA)
- T – TCPIP

Port

The port number; relevant for TCP/IP defined channels only.

This field is relevant only for sender channels. Receiver channels are started by

the MQSeries listener program which uses the port number configured in the global system definition.

Type

Channel type of sender, receiver, or client.

Partner

The channel partner name. This is the CICS connection ID for LU6.2 channels, or the IP address for TCP/IP channels.

For TCP/IP this field is relevant only for sender channels. Sender channels identify a specific host for communications, whereas receiver channels can accept communications from any host.

Allocation Retries – sender channels only

Number of Retries

The retry count field represents the number of times an allocation is retried when the conversation has not been established. You should set the retry count at less than 10. If this value is exceeded, the system can be placed under stress.

For receiver channels, this value should be set to zero.

Note: When you configure a new environment, failures occurring more frequently than this can indicate a network problem. You should investigate the problem LU, and its associated resources, to ensure that the session is bound and to establish why the conversation cannot be allocated.

Delay Time-Fast

The time interval, in seconds, that an allocation of conversation is retried for the first cycle of retries. A value of one to five seconds is sufficient for this field, with the longer time being used for a slow environment, for example, a dial-up SDLC.

For receiver channels, this value should be set to zero.

Delay Time-Slow

The time interval, in seconds, that an allocation of conversation is retried for the next cycle of retries, should the first cycle of retries fail. A value between three and 10 seconds is sufficient for this field, with the longer time being used for a slow environment.

For receiver channels, this value should be set to zero.

Get Retries – sender channels only

Number of Retries

The number of retries for the MQGET call when the queue is depleted. If a transmission queue is empty, the queue manager retries at the Delay-Time interval before disconnecting the channel or making a request to disconnect the channel.

For receiver channels, this value should be set to zero.

Delay Time

The time interval, in seconds, between retries. The value of this field may depend on the size of message and the platforms where the LU resides. The optimum value can vary from 1 to 20 seconds.

The longer the delay time specified, the less frequently a channel is reopened. For time-consuming dial-up connections, you are recommended to use a value of 20 seconds.

For receiver channels, this value should be set to zero.

Note: By using a value of zero for the Number of Retries, and a value of “n” seconds for the Delay Time it is possible for you to set a simple disconnection interval similar to that provided on other MQSeries platforms.

Max Messages per Batch

The maximum number of messages in the batch.

Message Sequence Wrap

The message sequence number (MSN) wrap count represents the highest MSN value used on this channel, after which the MSN reverts to one. You are recommended to set this value to 999 999.

Note: The value of the MSN Wrap count must be the same at both the sending and receiving ends of the channel.

Max Transmission Size

The mutually accepted maximum number of characters for each transmission. The minimum value should be equal to the maximum message size expected on this channel, plus 476 bytes for the transmission header.

Max Message Size

The maximum number of bytes for each message that is allowed for this channel.

TP Name

The remote task ID, character only, of the receiver on a remote CICS region or a Transaction Program name on a remote system. This is required by the sender, and since CICS uses four bytes as the transaction identifier, only the first four bytes of the remote task ID are meaningful for CICS to CICS conversation.

This field is not relevant for TCP/IP channels.

Note: VSE converts the name to uppercase, therefore, the corresponding name on the remote system should be defined in uppercase characters.

Checkpoint Frequency

A checkpoint event of this channel is taken after the specified input and output activities have occurred. The optimum value, in seconds, varies from 10 to 1000 depending on the emphasis of system throughput against channel recoverability.

Checkpoint Time Span

A checkpoint event of this channel is taken after the specified time interval, in seconds, has expired. You are recommended to use a value of 10 seconds as this does not present too much overhead.

Note: The Checkpoint Frequency and Checkpoint Time Span parameters are used for optimizing performance, where the link to remote systems is both stable and reliable. These parameters reduce the frequency with which MQSeries checkpoints the local MSN values.

Where the systems are not stable, or possible interruptions to message transmission are possible, you are recommended to set values of zero.

This ensures that MQSeries issues frequent checkpoints, and greatly reduces the possibility of MSNs getting out of sequence.

Queue configuration guidelines

When configuring the queue (see “Queue definitions” on page 48), use the following guidelines:

Physical File Name

The CICS file name, of up to seven characters, used to store messages for this queue. A physical file can hold as many queues as required. A message queue can be logically replenished, if its associated physical file name is changed.

Maximum Q Depth

The maximum number of records that can remain unread on this queue. Any inbound message that causes the queue depth to exceed this size is rejected as “Queue Full”. The minimum value you set should be the maximum number of messages on the queue before the application starts to read and process the queue. In practice, you can set this to 9,999,999.

Maximum Message Length

The maximum number of characters for each message that this queue allows. If this queue is a transmission queue, the value needs to be sufficiently large to accommodate all messages using this queue as the outbound queue.

Maximum Concurrent Accesses

The maximum number of MQOPEN calls that can occur on this queue. You are recommended to set a value of 100 for each queue that is not a transmission queue. For a transmission queue you should add a value of 100 calls, to the base of 100 calls, for each additional target queue that receives messages from this transmission queue. Setting a high value can use too much overhead.

Global Lock Entries

The maximum number of entries that the queue manager uses to maintain committed MQPUT and MQGET calls for this queue for system recovery. You should set this value to be equal to, or a little less than, the Maximum Number of Opens for this queue.

Local Lock Entries

The maximum number of entries that the queue manager uses to maintain uncommitted MQPUT and MQGET calls for this queue for recovery. Since an entry of a local lock is deleted once an application issues an explicit SYNCPOINT CICS command to commit updates, the more often an application takes the checkpoint, the fewer the maximum number of local locks needed. Specifying a value of 20 is usually sufficient.

Checkpoint Threshold

The maximum number of queue accesses between checkpoints to be taken by the queue manager for this queue. The smaller the value specified, the more often the queue manager takes checkpoints to secure the data integrity, thereby increasing the reliability. The larger the checkpoint threshold, the fewer the resources that are used by the queue manager. A value between 500 and 5000 is a good compromise between the requirements for performance and reliability.

Trigger Type

“F” is used to generate a trigger when an MQPUT call changes the status of a queue from empty to nonempty. The triggered transaction must have sufficient logic to empty the queue, including messages that may arrive during the

process, in a single thread. “E” is used to generate a trigger whenever an MQPUT call occurs and may have as many threads as specified in Max Trigger Starts.

Maximum Trigger Starts

The maximum number of trigger threads that can be activated simultaneously. This field applies to Trigger Type “E” only.

Trans ID

The transaction to be started by the trigger. This field is mutually exclusive with the Program ID. You are recommended to leave this field blank and use a Program ID, for example MQPSEND, unless you require a user transaction.

Program ID

You should use the MQPSEND call on a transmission queue if you require triggering.

Term ID

You should leave this field blank unless you require a terminal for problem determination purposes.

Permitted number of channels

The limit on the number of channels depends upon the availability of system resources. The queue manager can support as many channels and transmission queues as the resource in the system permits.

Example configuration

The following tables give a set of values that can be used to set up your system. See:

- Table 4 for the queue manager
- Table 5 on page 40 for a channel
- Table 6 on page 40 for a queue

Table 4. Example queue manager configuration

Parameter	Value	Units
Maximum Number of MQCONN	200	integer
Maximum Open Queue	200	integer
System Wait Interval	5	seconds
Maximum Q Depth	9999999	integer
Maximum Message Size	3345	bytes
Maximum Number of Opens	500	integer
Max Number of Global Locks	500	integer
Max Number of Local Locks	20	integer
Checkpoint Threshold	500	integer

Product configuration

<i>Table 5. Example channel configuration</i>		
Parameter	Value	Units
Allocation Retries	4	integer
Delay Time-Fast	1	second
Delay Time-Slow	3	seconds
Get Retries	1	integer
Delay Time	10	seconds
Message Sequence Wrap	999999	integer
Maximum Transmission Size	3821	bytes
Maximum Message Size	3345	bytes
Checkpoint Time Span	10	seconds

<i>Table 6. Example queue configuration</i>		
Parameter	Value	Units
Maximum Q Depth	999999	integer
Maximum Message Size	3345	bytes
Maximum Number of Opens	1000 (transmit queue) 100 (other queues)	integer
Max Number of Global Locks	1000 (transmit queue) 100 (other queues)	integer
Max Number of Local Locks	20	integer
Checkpoint Threshold	100	integer
Trigger Type	E	character
Maximum Trigger Starts	1	integer
Transaction Id	<blank>	character
Program Id	MQPSND	character

Chapter 4. System operation

There are two ways of managing an MQSeries for VSE/ESA system:

- You can use the CICS transaction MQMT.
MQMT allows you to configure, operate, and monitor an MQSeries for VSE/ESA system. MQMT also supports the browsing of message queues and is described in this chapter.
- You can use the MQSeries Command Line interface (MQCL).
MQCL supports management of queues and channels, and is described in Chapter 5, "MQSeries for VSE/ESA utility functions" on page 83.

MQSeries master terminal displays

The MQMT menus and display screens are organized in an *informal* hierarchy as depicted in the following diagram. The hierarchy is informal in the sense that non-hierarchical paths between screens can be invoked by using the function keys. For improved legibility, the chart omits certain exit and return paths available from lower level screens.

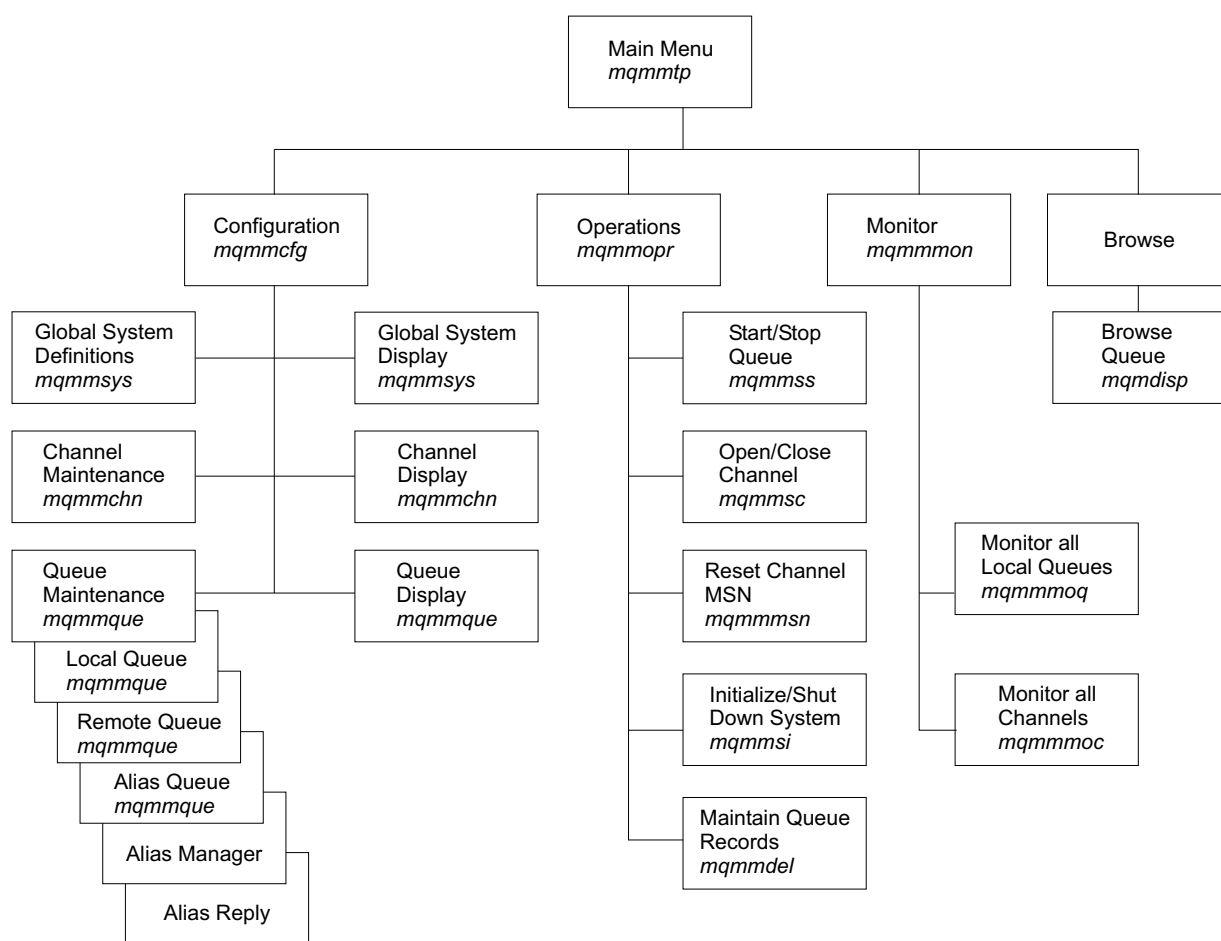


Figure 11. Display screen relationships

MQSeries master terminal

The main MQMT menu is shown in “MQSeries master terminal (MQMT) – main menu” on page 42, and the operator functions available through each of the secondary panels are shown in “Configuration functions” on page 44.

General panel layout

MQSeries panels are either menu panels or data entry panels. In either case, they show the following fields:

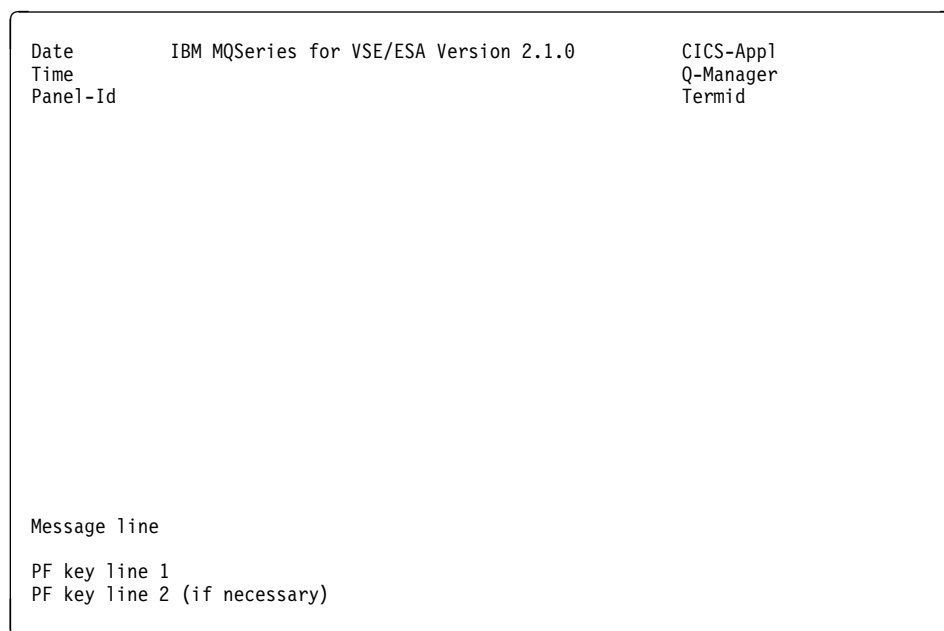


Figure 12. General panel layout

Where:

CICS-App1

The VTAM application ID for this CICS partition

Panel-Id

The name of the displayed panel

Q-Manager

The name of the MQSeries queue manager specified in the global definitions.

Termid

The ID of the CICS terminal on which this panel is displayed.

MQSeries master terminal (MQMT) – main menu

You can invoke the MQSeries system administrator program, MQMT, from any 3270 terminal. To access the operator functions, type MQMT at the CICS prompt.

When MQMT starts, the main menu is displayed.

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
07:55:22       *** Master Terminal Main Menu ***          VSE1
MQMMTP                                                A005

                SYSTEM IS ACTIVE

                1. Configuration
                2. Operations
                3. Monitoring
                4. Browse Queue Records

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
CLEAR/PF3 = Exit                            ENTER=Select

```

Figure 13. Master terminal main menu

From the main menu, one of several submenus can be selected. The first three selections correspond to broad categories that include most MQSeries operator functions:

- Configuring MQSeries
- Operating (controlling) MQSeries
- Monitoring MQSeries

The fourth function allows you to display the records on a selected queue:

- Browsing MQSeries queues

Each submenu presents a list of operator functions available from that screen. When a specific function is selected, the appropriate data entry or data display screens are presented to the operator.

Operator screen action keys

The action keys available on each MQSeries operator screen are displayed at the bottom of the screen with an explanation of their function. In general, the following keys are available and associated with the indicated action:

CLEAR	– Exit MQMT
PF2	– Return to previous menu
PF3	– Exit to CICS
PF4	– Select/Read (Same as Return or Enter keys)
PF5	– Add
PF6	– Update
PF7	– Backward
PF8	– Forward
PF9	– List
PF10	– Screen-dependant
PF12	– Delete

Configuration functions

Selecting option 1 (Configuration) from the master terminal main menu, (see Figure 13 on page 43) displays the following screen:

```
07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
07:56:10      *** Configuration Main Menu ***          VSE1
MQMMCFG                                             A005

                SYSTEM IS ACTIVE

                Maintenance Options :
                1. Global System Definition
                2. Queue Definitions
                3. Channel Definitions

                Display Options      :
                4. Global System Definition
                5. Queue Definitions
                6. Channel Definitions

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process      PF2 = Main Menu      PF3 = Quit
```

Figure 14. Configuration main menu

On this screen, selections 1, 2, and 3 allow you to perform maintenance functions on various MQSeries configuration objects. Selections 4, 5, and 6 allow viewing of the same objects.

Notes:

1. Changes to parameters on configuration screens take effect only when the queuing system is reinitialized.
2. When values are shown on the screens, they are default values.

Global system definition

Before you can do anything with messages and queues, you must create a queue manager. Once the installation process is complete, you use the MQSeries “Global System Definition” screen to create a queue manager and start it.

Default objects form the basis of any object definitions that you make. System objects are required for queue manager operation and you must create these objects for the queue manager that you created.

Guidelines for creating queue managers

A queue manager manages the resources associated with it, in particular the queues that it owns. It provides queuing services to applications for Message Queuing Interface (MQI) calls and commands to create, modify, display, and delete MQSeries objects. Some tasks you must consider when creating a queue manager are:

- Selecting a unique queue manager name, as described on Page 45.

- Creating a dead-letter queue, as described on Page 45.
- Backing up the configuration file, as described on Page 48.

The tasks in this list are explained in the sections that follow.

Specifying a unique queue manager name

When you create a queue manager, you must ensure that no other queue manager has the same name, anywhere in your network. Queue manager names are not checked at create time, and non-unique names will prevent you from using channels for distributed queuing.

One method of ensuring uniqueness is to prefix each queue manager name with its own (unique) node name. For example, if a node is called `accounts`, you could name your queue manager `accounts.saturn.queue.manager`, where `saturn` identifies a particular queue manager and `queue.manager` is an extension you can give to all queue managers. Alternatively, you can omit this, but note that `accounts.saturn` and `accounts.saturn.queue.manager` are *different* queue manager names.

If you are using MQSeries for communicating with other enterprises, you can also include your own enterprise as a prefix. We do not actually do this in the examples, because it makes them more difficult to follow.

Specifying a dead-letter queue

The dead-letter queue is a local queue where messages are put if they cannot be routed to their correct destination.

Attention: It is vitally important to have a dead-letter queue on each queue manager in your network. Failure to do so may mean that errors in application programs cause channels to be closed or that replies to administration commands are not received.

You create a dead-letter queue as a local queue; “Creating local queues” on page 49 for details.

For example, if an application attempts to put a message on a queue on another queue manager, but the wrong queue name is given, the channel is stopped, and the message remains on the transmission queue. Other applications cannot then use this channel for their messages.

The channels are not affected if the queue managers have dead-letter queues. The undelivered message is simply put on the dead-letter queue at the receiving end, leaving the channel and its transmission queue available.

Therefore, when you create a queue manager you should specify the name of the dead-letter queue.

Creating a queue manager

For each installation of the MQSeries system, one (and only one) queue manager must be defined. This is accomplished through the screen shown in Figure 15 on page 46. This screen is also used to modify previously defined global parameters.

Queue manager creation

```
07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
07:58:53        Global System Definition              VSE1
MQMMSYS         Queue Manager Information              A005
Queue Manager . . . . . : VSEP
Description Line 1 . . . . : VSE QUEUE MANAGER
Description Line 2 . . . . : ON SYSTEM WINVSEP
                Queue System Values
Maximum Number of Tasks . . : 00000100      System Wait Interval : 00000010
Maximum Concurrent Queues . . : 00000100      Max. Recovery Tasks : 0000
Allow TDQ Write on Errors : Y CSMT      Allow Internal Dump : Y
                Queue Maximum Values
Maximum Q Depth . . . . . : 01000000      Maximum Global Locks.: 00001000
Maximum Message Size. . . . : 01000000      Maximum Local Locks . : 00001000
Maximum Single Q Access . . : 00000100      Checkpoint Threshold : 1000
                Global QUEUE /File Names
Local Code Page . . . : 00500      TCP/IP Listener Port : 01414
Configuration File. : MQFCNFG      Licensed Clients . . : 00020
LOG Queue Name. . . : SYSTEM.LOG
Dead Letter Name. . : SYSTEM.EXCEPT
Monitor Queue Name. : SYSTEM.MONITOR

Requested record displayed.
PF2 = Main Config  PF3 = Quit  PF4/ENTER = Read  PF6 = Update
```

Figure 15. System queue manager information

On this screen the data entry fields are:

Queue Manager

This is the name of the local queue manager for this MQSeries system installation. The name may be up to 48 characters and must conform to the MQSeries naming requirements.

Description Lines 1 & 2

Text fields for operator use only. They may each be up to 32 characters.

Queue System Values

Maximum Number of Tasks

The maximum number of simultaneous connections to the queue manager.

Maximum Concurrent Queues

The maximum number of simultaneously open queues.

Allow TDQ Write on Errors

Y – allow writes to the CICS TDQ 'CSMT' if SYSTEM.LOG not available

N – do not allow write to the CICS TDQ

B – write to both SYSTEM.LOG and the CMST TDQ.

System Wait Interval

The sleep time in seconds for the system monitor program and start up of trigger programs after system initialization.

Max. Recovery Tasks

Maximum number of tasks attached by the system monitor when errors are detected in queues or control blocks attached to queues. A high number would lead to the use of too many CICS resources and have a negative impact on the overall CICS performance. The suggested value is zero.

Allow Internal Dump

Allow the MQSeries API to process a CICS Task Dump if the internal areas are corrupted.

Queue Maximum Values

Maximum Q Depth:

The maximum number of records that will be left unread on a queue.

Maximum Message Size

The maximum size of any message.

Maximum Single Q Access

The maximum number of object handles allowed for a queue.

Maximum Global Locks

The maximum number of entries that the queue manager uses to maintain destructive PUT or GET locks, per queue, for the system.

Maximum Local Locks

The maximum number of entries that the queue manager uses to maintain destructive PUT, or GET locks, per queue, for each individual task.

Checkpoint Threshold

The maximum number of queue accesses between checkpoints.

Global QUEUE /File Names

Local Code Page

The code page in use on the local system.

Configuration File

The CICS file definition name of the MQSeries configuration file.

LOG Queue Name

The queue name where the MQSeries programs write information and error messages.

Dead Letter Name

The file where channel programs write messages that are received with the wrong queue manager name or queue name. These messages will have the dead letter header placed in front of the queue record.

Monitor Queue Name

The queue that the API application requests when the system monitor is turned on.

TCP/IP Listener Port

The port number that MQSeries uses for accepting TCP/IP requests from remote queue managers and TCP/IP clients. The number reserved for MQSeries is 1414.

Licensed Clients

The number of clients for which this system is registered. Use the number from your MQSeries for VSE/ESA license documentation.

Note: Queue maximum value fields restrict the allowed values in the queue definition field values, while the rest of the fields affect the run-time values when the System is initialized.

Backing up the configuration file after creating the queue manager

When you create the queue manager, the queue manager configuration file MQFCNFG is updated. This contains configuration parameters for the queue manager, and the queue and channel definitions.

You should make a backup of this file. If you have to create another queue manager, perhaps to replace the existing queue manager if it is causing problems, you can reinstate the backup when you have removed the source of the problem.

Queue definitions

Selecting 2 on the configuration menu allows you to maintain (add, modify, or delete) queue definitions for the local installation of MQSeries.

Note: The same screens are used to accomplish the three functions of adding, modifying, or deleting a queue definition; the required action being selected through the function keys. The following sections present the screens that you see if you are adding a new queue definition.

“Modifying and deleting queue definitions” on page 59 explains how you modify or delete a queue.

To create a queue definition, multiple screens may be involved. The first screen is the same for all queues, and allows entry of the queue name and type.

Based on the type you enter, an appropriate second screen is displayed for you to enter the remainder of the data to complete the definition.

In the case of local queues, a third screen, the Extended Queue Definition Screen can be displayed.

The first screen displayed is as shown in Figure 16.

```
07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
08:01:30       Queue Main Options                      VSE1
MQMMQUE                                               A005

                SYSTEM IS ACTIVE

Default Q Manager : VSEP

Object Type:      L=Local Q, R=Remote Q, AQ=Alias Queue,
                  AM=Alias Manager,
                  AR=Alias Reply Q

Object Name:

ENTER NEEDED INFORMATION.

PF2=Main Config PF3 = Quit PF4/ENTER = Read   PF5 = Add      PF6 = Update
                  PF9 = List                   PF11= Reorg.  PF12= Delete
```

Figure 16. Queue main options screen

On this screen the data entry fields are:

Object Type

This is a two character field with the acceptable entries listed on the screen. The type determines the next screen to be displayed.

Object Name

This is the name of the queue (or alias) being defined. The name may be up to 48 characters, must be unique among all other defined queues for this installation, and must conform to the MQSeries naming requirements.

The Object Type you select on this screen is used to determine which of the definition screens is displayed:

- L selects a local queue definition; see “Creating local queues”
- R selects a remote queue definition; see “Create remote queue” on page 55
- AQ selects an alias queue definition; see “Create alias queue” on page 56
- AM selects an alias manager definition; see “Create alias queue manager” on page 57
- AR selects an alias reply queue definition; see “Create alias reply queue” on page 58

Creating local queues

Selecting “L” on Figure 16 on page 48 displays Figure 17.

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
08:02:20        Queue Definition Record              VSE1
MQMMQUE        QM - VSEP                              A005

                LOCAL QUEUE DEFINITION

Object Name. . . . . : ANYQ
Description line 1 . . . . . :
Description line 2 . . . . . :

Put Enabled . . . . . : Y      Y=Yes, N=No
Get Enabled . . . . . : Y      Y=Yes, N=No

Default Inbound status . . : A      Outbound .. : A      A=Active,I=Inactive
Dual Update Queue . . . . . :

Automatic Reorganize (Y/N) : N

Requested record displayed.

PF2=Main Config PF3 = Quit  PF4/ENTER = Read  PF5 = Add      PF6 = Update
                PF9 = List  PF10= Queue   PF11= Reorg.  PF12= Delete
    
```

Figure 17. Local queue definition

On this screen the data entry fields are:

Object Name

Filled in from the previous screen.

Description lines 1 & 2

Text fields for operator use only. They may each be up to 32 characters.

Put Enabled

A toggle that enables or disables MQPUT operations against this queue.

Get Enabled

A toggle that enables or disables MQGET operations against this queue.

Default Inbound status

Sets the initial status to active (A) or inactive (I) at run time for the inbound direction of the queue.

Outbound

Sets the initial status at run time for the outbound direction of the queue.

Dual Update Queue

When an existing queue name is entered here, dual queuing is activated. The queue being created becomes the primary queue, and the queue entered in this field becomes the dual queue. The definition of the dual queue is updated automatically with the name of the primary queue. The queue display of the dual queue has a corresponding heading "Dual Source Queue".

Dual Source Queue

The name of the primary queue, for which the queue being displayed is the dual queue. This field appears only when a local queue serves as a dual update queue.

Note: When an existing queue is defined as the dual to a primary queue, these two queues both participate in the same logical units of work.

If, for any reason, it becomes impossible to update the dual queue (for example, if the queue becomes disabled, the associated file is closed, or an ISC link is lost), updates continue to be made to the primary queue and the dual queue goes to a recovery status.

Automatic Reorganize

A toggle that enables or disables automatic reorganization of the VSAM file, associated with the selected queue, at specified time intervals.

Notes:

1. MQSeries for VSE/ESA uses VSAM files to move messages. The indexes of these files can become fragmented, causing the performance of the system to suffer.

To reorganize these files you must use VSAM utilities, and the Automatic Reorganize feature automates the process.

2. You are recommended to use the Automatic Reorganize feature only for queue files with high activity.
3. When you specify a queue file that is to be automatically reorganized, you should ensure that there is only one MQSeries queue associated with each VSAM file.
4. Ensure that you use the CICS system initialization parameter SPOOL=(YES..). Check with your CICS systems programmer to ensure that this is set correctly. See the *CICS for VSE/ESA System Definition and Operations Guide* for more information.

Local queue extended definition screen

By pressing function key 10 (PF10), you can display a second screen to enter the extended definition fields for the queue. In the case of a request to add a queue, this Extended Definition Screen is presented automatically. This detailed screen is shown in Figure 18:

```

07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
08:02:51           Queue Extended Definition                      VSE1
MQMMQUE           QM - VSEP                                       A005
Object Name. . . . . : ANYQ
                                     Physical Queue Information
Usage Mode . . . . . : N      N=Normal, T=Transmission
Share Mode . . . . . : Y      Y=Yes, N=No
Physical File Name . . . . : MQFI002  MQSERIES.TWO.DEVELOP.MQFI002
                                     Maximum Values
Maximum Q Depth. . . . . : 01000000  Global Lock Entries . : 00001000
Maximum Message Length . . : 01000000  Local Lock Entries. . : 00001000
Maximum Concurrent Accesses: 00000100  Checkpoint Threshold : 1000

                                     Trigger Information
Trigger Enable . . . . . : N      Y=yes, N=No
Trigger Type . . . . . :      F=First, E=Every
Maximum Trigger Starts . . : 0001
Allow Restart of Trigger : N      Y=Yes, N=No
Trans ID :
Program ID : MQPSEND
                                     Term ID:
                                     Channel Name:

Requested record displayed.
PF2=Main Config PF3 = Quit PF4/ENTER = Read  PF5 = Add          PF6 = Update
                PF9 = List PF10= Queue      PF11= Reorg.       PF12= Delete
    
```

Figure 18. Local queue extended definition

On this screen the data entry fields are:

Object Name

Filled in from the previous screen. Cannot be modified.

Local Queue Information

Usage Mode

Normal (N) means that the queue is used by an application to receive inbound messages. Transmission (T) means that the queue is used by MQSeries to hold outbound messages destined for another MQSeries queue manager.

Share Mode

Defines a queue as shareable or exclusive on input.

Physical File Name

The CICS file name, with a maximum of seven characters, used to store messages for this queue.

Maximum Values

Maximum Q Depth

The maximum number of messages allowed on this queue. The default value is the value specified in the global system definition.

Maximum Message Length

The maximum length of an application message processed on this queue.

Maximum Concurrent Accesses

The maximum number of MQOPEN calls that can occur for this queue simultaneously.

Creating local queues

Global Lock Entries

Allocates the locking table for this queue for all committed MQGET calls.

Local Lock Entries

This is used to allocate the locking table for this queue for each task's noncommitted MQGET calls.

Checkpoint Threshold

The maximum number of queue accesses between checkpoints.

Trigger Information

Trigger Enable

If you are defining a transmission queue for use with a sender channel, set this value to Yes (Y). Otherwise, for use with a server or receiver channel, set this field to No (N).

Trigger Type

F A trigger is generated when the first message arrives on an empty queue.

E A trigger is generated every nth message, where n is determined by the Maximum Trigger Starts field.

Only one transaction can be active against the queue if the Trigger Type is set to F.

Maximum Trigger Starts

The maximum number of trigger threads that can be active at once.

Allow Restart of Trigger

Allows the automatic restart of an application if the trigger count goes to zero. It restarts one trigger if messages are available on this queue.

Trans ID

The name of the transaction to be started by a trigger, with a length of four characters. If a transaction ID is specified, this transaction will be started. For a transmission queue, this field is left blank.

If a transaction identifier is defined, the program identifier should be left blank.

Program ID

The name of the user program to be invoked, with a length of eight characters.

If you are defining a transmission queue to be used with a sender channel, MQPSSEND must be used. If the field for Trans ID is left blank and this field contains a program ID, the specified program is linked.

Term ID

Optional field of four characters used for problem determination. It is attached to the transaction ID specified in the Trans ID field.

SYSID

Reserved.

Channel Name:

Is the channel name, with a maximum of 20 characters.

Notes:

1. The PF10 key can be used to toggle between the Local Queue Definition screen (Figure 17 on page 49) and the Local Queue Extended Definition screen (Figure 18 on page 51).
2. Either Trans ID or Program ID is required if triggering is enabled.
3. The internal MQSeries trigger API transaction MQ02 cannot be used as a trigger transaction ID.
4. Both a trigger transaction and a trigger program can be defined, but only the trigger transaction is activated and the trigger program name is passed in the trigger communications area.
5. The maximum message length is restricted by the global system maximum message size. The maximum message size cannot be bigger than the application message size plus the MQSeries header, and cannot be bigger than the VSAM CISIZE-7.

Navigation through the screens is dependent upon the PF keys.

Automatic reorganization

By pressing function key 11 (PF11), you can display a screen that enables automatic reorganization of the queue. The data entered on this screen creates a VSAM IDCAMS JCL stream. When submitted, this job recreates the VSAM queue file using the parameters entered.

The IDCAMS job is similar to the one used in the sample job MQJQUEUE.Z.

The queue manager waits until the queue is empty (queue depth=zero) and the correct start time reached. The queue manager then submits the job using the CICS SPOOL interface. When the job has finished, it communicates with the queue manager.

If the queue depth is not zero, the job will not be scheduled. This ensures that unprocessed messages are not deleted. In addition, the queue **must** be the only queue associated with this VSAM file. You are recommended to schedule these jobs for off-peak periods when the queue is likely to be inactive.

The detailed screen is shown in Figure 19 on page 54:

Creating local queues

```
06/03/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
12:10:56                               QM - VSEP                               VSE1
MQMMQUE                                                A003

VSAM Name. . . . . : MQSERIES.TWO.DEVELOP.MQFI001
VSAM Catalog . . . . : MQSERIES.TWO.PRODUCT.CATALOG
VSAM File. . . . . : MQFI001
VSE Volume . . . . . : VSEP22      Start Time : 0100
Records. . . . . : 005000      Interval : 0000
Avg. Record Size . . . . : 001024      Job Class : 0
Max. Record Size . . . . : 004089
CI Size. . . . . : 008192

Record updated OK.
PF2 = Options PF3 = Quit PF4/Enter = Read      PF6 = Update
PF7 = Down PF8 = Back      PF10= Queue      PF12= Delete
```

Figure 19. Automatic reorganization

On this screen the data entry fields are:

VSAM Name

VSAM cluster name.

VSAM Catalog

VSAM catalog in which the cluster is to be created.

VSAM File

VSAM file name.

VSE Volume

VSE volume name.

Records

The number of records for which space is reserved.

Avg. Record Size

IDCAMS average record size.

Max. Record Size

IDCAMS maximum record size.

CI Size

Data area control interval size.

Note: The index control interval size is fixed at 1024.

Start Time

Start time for the reorganization JCL to run.

0100 indicates that this job is scheduled to run at 01:00 AM.

Interval

Time in minutes for the next scheduled run after the start time.

Job Class

The VSE job class for the reorganization job.

Create remote queue

Selecting “R” on Figure 16 on page 48 displays Figure 20.

```

01/28/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:44:47       Queue Definition Record                  VSE1
MQMMQUE        QM - VSE2                                A005

                REMOTE QUEUE DEFINITION

Object Name. . . . . : CIC1/REMOTE
Description line 1 . . . . : Remote Queue for Testing the
Description line 2 . . . . : MQSeries/VSE System.

Put Enabled . . . . . : Y      Y=Yes, N=No
Get Enabled . . . . . : Y      Y=Yes, N=No

Default Inbound status . . . : A      Outbound .. : A      A=Active,I=Inactive

REMOTE QUEUE NAME . . . . . : VSE2
REMOTE QM NAME. . . . . : VSE2QM
TRANSMISSION Q NAME . . . . . : QUE.TEST

Record added OK.

PF2 = Options  PF3 = Quit   PF4/ENTER = Read  PF5 = Add   PF6 = Update
                PF9 = List   PF10= Extended  PF12= Delete
    
```

Figure 20. Remote queue definition

On this screen the data entry fields are:

Object Name

Filled in from the previous screen.

Description lines 1 & 2

Text fields for operator use only. They may each be up to 32 characters.

Put Enabled

A toggle that enables or disables MQPUT operations against this queue.

Get Enabled

A toggle that enables or disables MQGET operations against this queue.

Default Inbound status

Sets the initial status to active (A) or inactive (I) at run time for the inbound direction of the queue.

Outbound status

Sets the initial status at run time for the outbound direction of the queue.

REMOTE QUEUE NAME

The queue name on the remote MQSeries system to which the definition in progress will refer.

REMOTE QM NAME

The name of the remote MQSeries system queue manager on which the remote queue is defined as a local queue. This name must be defined as a local transmission queue unless the Transmission Q Name field is used.

TRANSMISSION Q NAME

The name of the local transmission queue to be used by MQSeries to convey messages to this remote queue. If the field is left blank, the remote queue manager name is required to map to a local transmission queue.

Creating alias queues

Note: Some other operating systems, with which you could be communicating, may be case sensitive. You should read the information in “Uppercase translation” on page 15 before devising a name for a queue, channel or queue manager.

Navigation through the screens is dependent upon the PF keys.

Create alias queue

Selecting “AQ” on Figure 16 on page 48 displays Figure 21.

```
01/28/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:47:16       Queue Definition Record                VSE1
MQMMQUE        QM - VSE2                               A005

                ALIAS QUEUE DEFINITION

Object Name. . . . . : QTST
Description line 1 . . . . : Alias queue name for QUE.TEST
Description line 2 . . . . :

Put Enabled . . . . . : Y      Y=Yes, N=No
Get Enabled . . . . . : Y      Y=Yes, N=No

Default Inbound status . . : A      Outbound .. : A      A=Active,I=Inactive

ALIAS QUEUE NAME. . . . . : QUE.TEST

Record added OK.

PF2 = Options  PF3 = Quit   PF4/ENTER = Read  PF5 = Add   PF6 = Update
                PF9 = List   PF10= Extended  PF12= Delete
```

Figure 21. Alias queue definition

On this screen the data entry fields are:

Object Name

Filled in from the previous screen.

Description lines 1 & 2

Text fields for operator use only. They may each be up to 32 characters.

Put Enabled

A toggle that enables or disables MQPUT operations against this queue.

Get Enabled

A toggle that enables or disables MQGET operations against this queue.

Default Inbound status

Sets the initial status to active (A) or inactive (I) at run time for the inbound direction of the queue.

Outbound status

Sets the initial status at run time for the outbound direction of the queue.

Alias Queue Name

The name of another object already defined in the local configuration. This must be a local queue name. It cannot identify another alias.

Navigation through the screens is dependent upon the PF keys.

Create alias queue manager

Selecting "AM" on Figure 16 on page 48 displays Figure 22.

```

01/28/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:48:28       Queue Definition Record                   VSE1
MQMMQUE        QM - VSE2                                 A005

                ALIAS MANAGER DEFINITION

Object Name. . . . . : VSE2QM
Description line 1 . . . . : Alias for Queue Manager
Description line 2 . . . . : VSE2

Put Enabled . . . . . : Y      Y=Yes, N=No
Get Enabled . . . . . : Y      Y=Yes, N=No

Default Inbound status . . : A      Outbound .. : A      A=Active,I=Inactive

ALIAS QM NAME . . . . . : VSE2
TRANSMISSION QUEUE. . . . . :

Record added OK.

PF2 = Options  PF3 = Quit   PF4/ENTER = Read  PF5 = Add   PF6 = Update
                PF9 = List   PF10= Extended  PF12= Delete
    
```

Figure 22. Alias queue manager definition

On this screen the definitions cannot be used in an MQCONN call; they may be used for MQOPEN substitution only. The data entry fields are:

Object Name

Filled in from the previous screen.

Description lines 1 & 2

Text fields for operator use only. They may each be up to 32 characters.

Put Enabled

A toggle that enables or disables MQPUT operations against this queue.

Note: You cannot enter data in this field.

Get Enabled

A toggle that enables or disables MQGET operations against this queue.

Note: You cannot enter data in this field.

Default Inbound status

Sets the initial status to active (A) or inactive (I) at run time for the inbound direction of the queue.

Outbound status

Sets the initial status at run time for the outbound direction of the queue.

Alias QM Name

The name of a known queue manager. This can be a local transmission queue name, a remote queue manager name, or the local queue manager name. It cannot identify another alias.

Transmission Queue

The name of the local transmission queue to be used by MQSeries to convey messages to this remote queue manager. If this field is left blank, the Alias QM

Creating alias reply queues

Name field is required to map to a local transmission queue or to the local queue manager name.

Navigation through the screens is dependent upon the PF keys.

Create alias reply queue

Selecting "AR" on Figure 16 on page 48 displays Figure 23.

```
01/28/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:49:45      Queue Definition Record          VSE1
MQMMQUE       QM - VSE2                          A005

                ALIAS REPLY DEFINITION

Object Name. . . . . : REPLYQ
Description line 1 . . . . : Alias Reply Definition
Description line 2 . . . . :

Put Enabled . . . . . : Y      Y=Yes, N=No
Get Enabled . . . . . : Y      Y=Yes, N=No

Default Inbound status . . : A      Outbound .. : A      A=Active,I=Inactive

ALIAS QUEUE NAME. . . . . : QUE.TEST
ALIAS QM NAME . . . . . : VSE2QM

Record added OK.

PF2 = Options  PF3 = Quit    PF4/ENTER = Read  PF5 = Add   PF6 = Update
                PF9 = List    PF10= Extended   PF12= Delete
```

Figure 23. Alias queue reply definition

On this screen, the definitions cannot be used in the MQOPEN call; they may only be used for reply queue name substitution with a MQPUT call. The data entry fields are:

Object Name

Filled in from the previous screen.

Description lines 1 & 2

Text fields for operator use only. They may each be up to 32 characters.

Put Enabled

A toggle that enables or disables MQPUT operations against this queue.

Note: You cannot enter data in this field.

Get Enabled

A toggle that enables or disables MQGET operations against this queue.

Note: You cannot enter data in this field.

Default Inbound status

Sets the initial status to active (A) or inactive (I) at run time for the inbound direction of the queue.

Outbound status

Sets the initial status at run time for the outbound direction of the queue.

Alias Queue Name

The name of another object already defined in the local configuration. This can be a local queue name or a remote queue name. It cannot identify another alias.

Alias QM Name

The name of a known queue manager. This can be a local transmission queue name, a remote queue manager name, or the local queue manager name. It cannot identify another alias.

Navigation through the screens is dependent upon the PF keys.

Modifying and deleting queue definitions

Selecting 2 on the configuration menu also allows you to modify, or delete queue definitions.

Note: You use the same primary screens for modifying, deleting, or adding a queue.

Selecting an existing queue definition

To modify or delete an existing queue definition, you must select the definition on which to work, and display it.

To do this, select option 2 on the “Configuration Main Menu” screen to display the “Queue Main Options” screen (see Figure 16 on page 48) and use either the PF4, or PF9, function key.

PF4 is the Read key, and you use it to bring a specific queue definition to the screen as follows:

1. Enter the name of the desired queue in the Object Name field.
2. Press PF4 or Enter.
3. MQSeries reads and displays the selected queue definition.

PF9 is the **LIST** key, and you use it to bring a specific queue definition to the screen as follows:

1. Press PF9.
2. The MQSeries System displays a list of all defined queues (see Figure 24 on page 60).
3. Select the desired queue by typing an “X” next to its name, or by placing the cursor on the appropriate object.
4. Press PF4 or Enter.
5. MQSeries reads and displays the selected queue definition.

Channel definitions

```
07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
11:50:08           Object List Screen                             VSE2
MQMMQUE                                                    0002

S Object                                                    Type
x
CIC1/REMOTE                                               Remote Queue
GEGE                                                       Local Queue
LONGQ                                                      Local Queue
MVS1_LOCAL                                                 Local Queue
MVS1_REMOTE                                               Remote Queue
MVS1_TQ                                                    Local Queue
QL.DEVL                                                    Remote Queue
QL.DEVL.X                                                 Local Queue
QR.OS2                                                     Remote Queue
...More

Records found          - Select one object name.

PF2 = Options   PF3 = Quit   PF4/ENTER= Read   PF5 = Add   PF6 = Update
                PF7 = Up     PF8 = Down      PF12 = Delete
```

Figure 24. Object list screen

Modifying an existing queue definition

When you have displayed the required queue definition, as described in “Selecting an existing queue definition” on page 59, you can modify any field in the definition. This may involve multiple screens to include all fields of the queue definition – see “Queue definitions” on page 48.

When you have made the changes you need, update the screen using the PF6 (UPDATE) function key.

Deleting an existing queue definition

When you have displayed the required queue definition, as described in “Selecting an existing queue definition” on page 59, you can delete it using the PF12 (DELETE) function key.

You will be asked to confirm that you want to delete the queue definition. You must press the PF12 function key again to delete the queue.

Channel definitions

Selecting 3 on the configuration menu allows you to maintain (add, modify, or delete) channel definitions for the local installation of the MQSeries System.

Note: The same screens are used to accomplish the three functions of adding, modifying, or deleting a channel definition; the required action being selected through the function keys. The following sections present the screens that you see if you are adding a new channel definition.

“Modifying and deleting channel definitions” on page 63 explains how you modify or delete a queue.

The screen shown in Figure 25 on page 61 is displayed to create a channel definition:


```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
08:03:53        Channel Record      DISPLAY      MCHN
MQMMCHN      Last Check Point      Last Update 00000000      A005
MSN 00000002 Time 07:10:22 Interv 000000      Create Date 19980528
Name : SD01_TCP_VSEP
Protocol : L (L/T) Port : 1414 Type : R (S/R/C)
Partner :

      Allocation Retries      Get Retries
      Number of Retries: 00000000      Number of Retries : 00000000
      Delay Time - fast: 00000000      Delay Time : 00000000
      Delay Time - slow: 00000000

      Max Messages per Batch : 000001      Max Transmission Size : 032000
      Message Sequence Wrap : 999999      Max Message Size : 008192

      Mess Seq Req(Y/N): Y      Convers Cap (Y/N): Y      Split Mssg(Y/N): N

Transmission Queue Name :
TP Name:
Checkpoint Values:      Frequency: 0000      Time Span: 0000
Enable(Y/N) Y      Dead Letter Store(Y/N) N
Channel record displayed.
PF2 =Menu PF3 =Quit PF4 =Read PF5 =Add PF6=Update PF9 =List PF12 =Delete

```

Figure 25. Channel record

On this screen the data entry fields are:

Name

The name of the channel to be defined.

Protocol

The protocol being used by the selected channel, which can be LU 6.2, or TCP/IP

Port

The port number that MQSeries uses for accepting TCP/IP requests from remote queue managers and TCP/IP clients. The number reserved for MQSeries is 1414.

Type

S: A sender only channel.

R: A receiver only channel.

C: A client channel

Requester Channels are not supported for IBM MQSeries for VSE/ESA.

Partner

The name of the LU6.2 connection, or the TCP/IP IP address.

Allocation Retries

Number of Retries

Number of allocation retries when not successful.

Delay Time - fast

Time between retries (in seconds).

Delay Time - slow

Time between retries (in seconds) after fast number of retries has been depleted.

Get Retries

Number of Retries

The number of Get retries when queue is empty.

Delay time

The time between retries (in seconds).

Channel negotiation fields

Max Messages per Batch

The mutually accepted maximum number of messages per batch to be transmitted.

Message Sequence Wrap

The mutually agreed maximum messages count before the count sequence starts over.

Max Transmission Size

The mutually accepted maximum number of bytes per transmission.

Max Message Size

The mutually accepted maximum number of bytes per message. The Maximum Message Size must be bigger than the application message size, plus approximately 1,000 bytes for the MQSeries header.

Mess Seq Reqd

This **must** be set to Y for MQSeries for VSE/ESA Both ends of the channel must use message sequence numbers.

Convers Cap

This is used by MQSeries to determine the translation required for message headers between various hardware platforms on the network. MQSeries for VSE/ESA is capable of translating message headers, and this **must** be set to Y.

Split Mssg

This **must** be set to Y if messages longer than the maximum transmission size for the channel are to be sent across the channel.

Other channel data

Connection ID

A four-byte field identifying the connection. Required by the sender, not recognized for the receiver.

Transmission Queue Name

The name of the transmission queue. Required for the sender, optional for the receiver.

TP Name

A 64-character field identifying the remote task ID of the receiver on the remote CICS region, or a TPNAME on the remote system (for example, AMQCRS6A). This is required by the sender for LU6.2 use only. This field is not recognized for TCP/IP.

Note: Although the TPNAME can be up to 64 bytes in other MQSeries products, on MQSeries for VSE/ESA it can have a maximum of 4 bytes only.

Checkpoint values

Parameters that effect how often synchronization is performed for this channel.

Frequency

Determines checkpoint event based upon I/O frequency.

Time Span

Determines checkpoint event based upon time span in seconds.

Enable

Enable the channel for communications at initialization time.

Dead Letter Store

Allow messages for undefined destinations to be written to the dead letter queue.

Modifying and deleting channel definitions

Selecting 3 on the configuration menu also allows you to modify, or delete channel definitions.

Note: You use the same primary screens for modifying, deleting, or adding a channel.

Selecting an existing channel definition

To modify or delete an existing channel definition, you must select the definition on which to work, and display it.

To do this, select option 3 on the “Configuration Main Menu” screen to display the “Channel Record” screen (see Figure 25 on page 61), and use either the PF4 or PF9 function key.

PF4 is the Read key, and you use it to bring a specific channel definition to the screen as follows:

1. Enter the name of the desired channel in the Channel Name field.
2. Press PF4 or Enter.
3. MQSeries reads and displays the selected channel definition.

PF9 is the List key, and you use it to bring a specific channel definition to the screen as follows:

1. Press PF9.
2. MQSeries displays a list of all defined channels (see Figure 26 on page 64).
3. Select the desired channel by typing an “S” next to the channel name.
4. Press PF4 or Enter.
5. MQSeries reads and displays the selected channel definition.

Modifying channel definitions

```
07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:52:01              Channel List                      VSE2
MQMMCHN                                               QMMC

S   CHANNEL NAME      TYPE  STATUS  LAST MSN  LAST CHECKPOINT
C1
MQM7.LU62.VSEF.DEVL  R   ENABLE  000009   10:28:31
MVS1_TO_VSE2        R   ENABLE  000005   10:28:31
VSE1_TO_VSE2        R   ENABLE  000009   10:28:31
VSE2_TO_MVS1        S   ENABLE  000069   10:28:31      MVS2
VSE2_TO_SD01        S   ENABLE  000006   11:11:09      SD01
VSE2_TO_VSE1        S   ENABLE  000000   10:28:31      VSE1

ENTER 'S' to select Channel
PF2 = Menu      PF3 = Quit      PF4 = Read      F7 = Backward  PF8 = Forward
```

Figure 26. Channel list

On this screen the display fields are:

Channel Name

The names of all channels.

Type

Type is sender (S) or receiver (R).

Status

Channel may be enabled (ENABLE) or disabled (DISABL).

Last MSN

The last checkpointed message sequence number of the channel.

Last Checkpoint

The time of the last checkpoint.

Modifying an existing channel definition

When you have displayed the required channel definition, as described in “Selecting an existing channel definition” on page 63, you can modify any field in the definition.

When you have made the changes you need, update the screen using the PF6 (Update) function key.

Deleting an existing channel definition

When you have displayed the required channel definition, as described in “Selecting an existing channel definition” on page 63, you can delete it using the PF12 (Delete) function key.

You will be asked to confirm that you want to delete the channel definition. You must press the PF12 function key again to delete the channel.

Global system definition display

Selecting 4 on the main menu allows you to view the attributes defined for the local queue manager, and all system-wide parameters, through the screen shown in Figure 27, which is display only:

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
07:58:53              Global System Definition              VSE1
MQMMSYS              Queue Manager Information              A005
Queue Manager . . . . .: VSEP
Description Line 1. . . . .: VSE QUEUE MANAGER
Description Line 2. . . . .: ON SYSTEM WINVSEP
                    Queue System Values
Maximum Number of Tasks . . . . .: 00000100      System Wait Interval : 00000010
Maximum Concurrent Queues . . . . .: 00000100      Max. Recovery Tasks  : 0000
Allow TDQ Write on Errors : Y   CSMT      Allow Internal Dump  : Y
                    Queue Maximum Values
Maximum Q Depth . . . . .: 01000000      Maximum Global Locks.: 00001000
Maximum Message Size. . . . .: 01000000      Maximum Local Locks  : 00001000
Maximum Single Q Access . . . . .: 00000100      Checkpoint Threshold : 1000
                    Global QUEUE /File Names
Local Code Page . . . . .: 00500      TCP/IP Listener Port : 01414
Configuration File. . . . .: MQFCNFG      Licensed Clients . . . : 00020
LOG Queue Name. . . . .: SYSTEM.LOG
Dead Letter Name. . . . .: SYSTEM.EXCEPT
Monitor Queue Name. . . . .: SYSTEM.MONITOR

Requested record displayed.
PF2 = Main Config   PF3 = Quit   PF4/ENTER = Read   PF6 = Update

```

Figure 27. Global system definition display

To return to the Configuration Main Menu, press the PF2 key.

Queue definition display

Selecting 5 on the main menu allows you to view existing queue definitions.

Note: This function allows you to see the queue definition, not the current queue status. To see the current queue information, see “Monitor queues” on page 75.

This operation is identical to the modify queue and delete queue operations (as described in “Modifying and deleting queue definitions” on page 59) except that the maintenance function keys, PF5 (Add), PF6 (Update), and PF12 (Delete), are not available.

Channel definition display

Selecting 6 on the main menu allows you to view existing channel definitions.

Note: This function allows you to see the channel definition, not the current channel status. To see the current channel information, see “Monitor channel” on page 78.

This operation is identical to the modify channel and delete channel operations (as described in “Modifying and deleting channel definitions” on page 63) except that the maintenance function keys, PF6 (Update), and PF12 (Delete), are not available.

Operations functions

Selecting option 2 (Operations) from the master terminal main menu (see Figure 13 on page 43) displays the following screen:

```
07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:53:32        *** Operations Main Menu ***          VSE1
MQMMOPR                                               A005

                SYSTEM IS ACTIVE

                1. Start / Stop Queue(s)
                2. Open / Close Channel(s)
                3. Reset Message Sequence Number
                4. Initialization / Shutdown of System
                5. Maintain Queue Message Records

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process                            PF2 = Main Menu      PF3 = Quit
```

Figure 28. Operations main menu

On this screen, selections correspond to available operator control functions.

Start/Stop queue

Selection 1 on the Operations menu allows you to start or stop processing for a queue.

This differs from setting the queue's Get Enabled or Put Enabled option to "No" in that the Start and Stop functions:

- Are dynamic, with immediate effects
- Apply universally to all processes attempting to access a local queue

The Get Enabled and Put Enabled functions are:

- Static configuration fields that take effect at system initialization time, or through the "Refresh from Config" option on the screen shown in Figure 29 on page 67.
- Can be selectively applied to aliases and remote queue definitions.

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
08:07:19       Start / Stop Queue                          VSE1
MQMMSS                                                A005

                System Information
System Status   : SYSTEM IS ACTIVE
Queue Status    : Queuing System is active.
Channel Status  : Channel System is active.
Monitor Status  : Monitor is not active.
                Single Queue Request
Queue Name      : QL.VSEP
Function        : x      S=Start, X=Stop, R=Refresh from Config
Mode            : b      I=Inbound, O=Outbound, B=Both

INBOUND Status :          DISABLED
OUTBOUND Status:          DISABLED

                All Queues Request
Function        :          S=Start, X=Stop, or M=Monitor

QUEUE STOPPED.
ENTER=Display   PF2 = Oper Menu   PF3 = Exit       PF6 = Update

```

Figure 29. Start / Stop queue control screen

On the Start / Stop Queue screen shown in Figure 29, the fields are:

System Information

System Status

Reflects the status of the system. This is normally ACTIVE, unless the system has not been initialized, or has been shutdown. When this occurs the field reads SYSTEM IS SHUTDOWN.

Queue Status

Reflects the status of the queuing system. This is normally ACTIVE, unless the system has not yet been initialized or all the queues have been stopped. When this occurs the field reads QUEUING SYSTEM IS STOPPED.

Channel System

Reflects the status of the channels. This is normally ACTIVE, unless the system has not yet been initialized or all the channels have been closed. When this occurs the field reads CHANNEL SYSTEM IS CLOSED.

Monitor Status

Reflects the status of the system monitor.

Single Queue Request

Queue Name

The name of a specific queue to start or stop

Function

The function to be performed, as follows:

S is to start:

- A stopped local queue
- The associated trigger mechanism
- Receiving messages if the channel is open

X is to stop a local queue and make it unavailable.

Operations functions

R is to refresh the run-time information for this queue from the configuration file, which was updated either by checkpoint requests or MQMT queue configuration. The configuration file (MQFCNFG) contains definitions of the queue manager, channels and queues.

Note: It is important to refresh a queue if the definition is changed, otherwise the change does not come into effect until the next initialization of the queue manager.

Mode

The queue process to be operated on, as indicated on screen.

INBOUND Status

Reflects the status of the specified queue. This is normally ACTIVE or IDLE unless the queue inbound has been stopped. If the queue is stopped, DISABLED is also displayed.

OUTBOUND Status

Reflects the status of the specified queue. This is normally ACTIVE or IDLE unless the queue outbound has been stopped. If the queue is stopped, DISABLED is also displayed.

All Queues Request

Function

The function to be performed, as follows:

S is to start the system queue manager without affecting system resources.

X is to stop the system queue manager without affecting system resources.

Note: When a queue manager is initiated or shut down, there are certain system functions that have to be performed to ensure that the contents of the disk files are in synchronization with the storage control blocks. However, when a system is started or stopped, the queue manager is simply enabled or disabled.

M toggles the monitor flag. This flag is used to log application requests and their results to the system monitor queue.

Notes:

1. Only local queues can be stopped or started. In order to stop or start a queue that is not local, the queue definition must be updated in the Put-Enabled or Get-Enabled fields. These configuration changes must then be "refreshed" to the run-time environment.

2. When a local queue is started, any associated triggers will also be started, if the queue depth reflects that messages are present.

This does not happen when an "All Queues Request" function is performed. Additionally, any queues that were stopped before the "All Queues Request" stop function was performed, remain stopped when the corresponding start function is performed.

Use the Monitor Queue function to check which local queues are stopped.

3. If the queue definition specifies a trigger and a sender channel, then starting a queue triggers the sender program to activate the channel and transmit messages.

Open / Close channel

Selecting 2 on the Operations main menu (see Figure 28 on page 66) allows you to open or close communications on an existing channel.

Note: Opening or closing a channel is not the same as starting or stopping the MCA process. See “Communications process” on page 81 for further information.

Figure 30 shows the first screen to be displayed:

```

07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
08:08:13           Open / Close Channel                          VSE1
MQMMS C                                                    A005

                System Information
System Status    : SYSTEM IS ACTIVE
Queue Status     : QUEUING SYSTEM IS ACTIVE
Channel System   : CHANNEL SYSTEM IS ACTIVE
                Single Channel Request
Channel Name     : SD01_LU62_VSEP

Function        : 0      0=Open , C=Close , R=Refresh from Config
Status          :

                All Channels Request
Function        :      0=Open , C=Close

CHANNEL HAS BEEN OPENED.
ENTER= Display  PF2 = Oper Menu      PF3 = Exit      PF6 = Update

```

Figure 30. Open / Close Channel

On this screen the fields are:

System Information

System Status

Reflects the status of the system. This is normally ACTIVE, unless the system has not been initialized, or has been shutdown. When this occurs the field reads SYSTEM IS SHUTDOWN.

Queue Status

Reflects the status of the queuing system. This is normally ACTIVE, unless the system has not yet been initialized or all the queues have been stopped. When this occurs the field reads QUEUING SYSTEM IS STOPPED.

Channel System

Reflects the status of the channels. This is normally ACTIVE, unless the system has not yet been initialized or all the channels have been closed. When this occurs the field reads CHANNEL SYSTEM IS CLOSED.

Single Channel Request

Channel Name

The name of a specific channel to start or stop

Function

The function to be performed:

- O is to open a closed channel.
- C is to close an open channel.
- R is to restore the run-time information from the configuration file. A channel must be refreshed if the definition was updated by MQMT channel configuration.

Status

Reflects the status of the specified channel. This is normally ACTIVE or IDLE unless the channel has been stopped. In this situation DISABLED is also displayed.

All Channels Request

Function

This either opens or closes the channel system.

Note: Opening a channel does not cause a trigger to activate. However, starting the channel's transmission queue does activate a trigger; see Step 3 on page 68.

Reset message sequence number

Selecting 3 on the Operations Main Menu (see Figure 28 on page 66) allows you to reset the message sequence numbers on an existing channel by displaying the screen shown in Figure 31:

```
07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
08:09:51           Reset Channel Message Sequence                VSE1
MQMMSN                                                     A005

                        System Information
System Status       : SYSTEM IS ACTIVE
Queue Status       : QUEUING SYSTEM IS ACTIVE
Channel Status     : CHANNEL SYSTEM IS ACTIVE
                        Reset Channel Info
Channel Name       : SD01_LU62_VSEP

Status              : IDLE

Current Next-MSN   : 00000102
New Next-MSN      : 00000099

Information updated.
PF2 = Oper Main Menu      PF3 = Cancel      PF6 = Update
```

Figure 31. Reset channel message sequence

On this screen the fields are:

System Information

System Status

Reflects the status of the system. This is normally ACTIVE, unless the system has not been initialized, or has been shutdown. When this occurs the field reads SYSTEM IS SHUTDOWN.

Queue Status

Reflects the status of the queuing system. This is normally ACTIVE, unless the system has not yet been initialized or all the queues have been stopped. When this occurs the field reads QUEUING SYSTEM IS STOPPED.

Channel System

Reflects the status of the channels. This is normally ACTIVE, unless the system has not yet been initialized or all the channels have been closed. When this occurs the field reads CHANNEL SYSTEM IS CLOSED.

Reset Channel Info

Channel Name

The name of a specific channel.

Status

Reflects the status of the specified channel. This is normally ACTIVE or IDLE unless the channel has been stopped. In this situation DISABLED is also displayed.

Current Next-MSN

Displays the message sequence number to be used next.

New Next-MSN

Operator-entered value for message sequence number to be used next.

Note: In order for a channel message sequence number to be reset, the channel must be stopped.

Initialization of system

Selecting 4 on the operations menu allows you to initialize the queuing system by displaying the screen shown in Figure 32:

```
07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
09:01:28           Initialization / Shutdown of System            VSE1
MQMMSI                                                     A005

                System Information
System Status    : SYSTEM IS ACTIVE
Queue Status     : QUEUING SYSTEM IS ACTIVE
Channel Status   : CHANNEL SYSTEM IS ACTIVE

Function        :          I=Initialize, X=Shutdown

Returned Results :

System initialized on  07/16/1998 at 08:14:26

Please enter one of the options listed.

PF2 - Main Operation          PF3 - Cancel          PF6 - Update
```

Figure 32. Initialization of system

On this screen the fields are:

System Information

System Status

Reflects the status of the system. This is normally ACTIVE, unless the system has not been initialized, or has been shutdown. When this occurs the field reads SYSTEM IS SHUTDOWN.

Queue Status

Reflects the status of the queuing system. This is normally ACTIVE, unless the system has not yet been initialized or all the queues have been stopped. When this occurs the field reads QUEUING SYSTEM IS STOPPED.

Channel System

Reflects the status of the channels. This is normally ACTIVE, unless the system has not yet been initialized or all the channels have been closed. When this occurs the field reads CHANNEL SYSTEM IS CLOSED.

Function

The function to be performed:

- I is to initialize the system. If the system is initialized, the queue manager is started and all channels and queues opened. Any trigger associated with queues just initialized is also activated if the queue depth is nonzero.
- X is to shutdown the system. If the system is shutdown, the queue manager is stopped and all the channels closed.

Returned Results

Pressing PF6 with an Initialize function (I) on this screen causes the static system configuration files to be loaded into the CICS/VSE dynamic storage. Any error messages or progress messages are displayed in this field.

Note: All outstanding queue maintenance requests must have completed before you perform an initialize or shutdown operation.

Queue maintenance

Selecting 5 on the operations menu allows you either to reset deleted records or physically delete records, by displaying the screen shown in Figure 33:

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
09:02:43       Maintain Queue Message Records             VSE1
MQMMDEL                                               A005

                System Information
System Status   : System is active.
Queue Status    : Queuing system is active.
Channel System  : Channel system is active.
                Queue Information
Queue Name      : ANYQ
Function        : A A=Delete all, D=Delete to date/time exclusive
                  R=Reset from date/time inclusive
Date (yyyymmdd) :
Time (hhmmss)  :

                Results of Request
Number Processed : 00000016
Number of Bypass : 00000000
New Last Read QSN: 00000000
Process Time     : 00:00:00

Queue processing finished.
PF2 = Oper Main Menu      PF3 = Quit
                          PF6 = Update
                          PF12= Retry

```

Figure 33. Maintain Queue Message Records

On this screen the fields are:

System Information

System Status

Reflects the status of the system. This is normally ACTIVE, unless the system has not been initialized, or has been shutdown. When this occurs the field reads SYSTEM IS SHUTDOWN.

Queue Status

Reflects the status of the queuing system. This is normally ACTIVE, unless the system has not yet been initialized or all the queues have been stopped. When this occurs the field reads QUEUING SYSTEM IS STOPPED.

Channel System

Reflects the status of the channels. This is normally ACTIVE, unless the system has not yet been initialized or all the channels have been closed. When this occurs the field reads CHANNEL SYSTEM IS CLOSED.

Queue Information

Queue Name

The name of the local queue on which the function is to be performed.

Function

The function to be performed:

- D is to delete messages that have been logically deleted up to a specified "written" exclusive date and time. For example, given the date and time of

Operations functions

980227230000, specifying “D” deletes all records with a written time prior to 11:00:00 p.m.

Note: Specifying D does not actually reclaim VSAM space, because record keys are always created in ascending sequence. You are strongly recommended to read “VSAM file maintenance” on page 90 for information regarding the Delete All function in relation to VSAM files.

- A is to delete all records (logically deleted, or written) and reclaim VSAM space
- R is to reset all logically deleted records to written status from a specified “deleted” inclusive date and time. For example, given the date and time of 980227230000, specifying “R” resets all delivered messages with delivery time after 10:59:59 p.m.

Date

The last date up to which the selected function is to be performed, if applicable.

Time

The last time up to which the selected function is to be performed, if applicable.

Results of request

Number Processed	Number of messages processed
Number of Bypass	Number of messages bypassed
New Last Read QSN	Last read queue sequence number
Process Time	Time to process the request

You use the PF6 (Update) function key to activate the requested function. The function itself is performed by another task, which signals the screen when it is complete. To display the signal, press the Enter key.

The PF12 (Retry) function key is used only if the delete task has ended before finishing the current request, and acts as a new PF6 request.

Notes:

1. A Delete or Reset Messages by Date and Time performs the requested function up to the selected date and time, but does not include records with that date and time.
2. If the queue is examined with the browse function, the put time of the last message to be reset should be the value for date and time.
3. The Delete All function purges all records, which include both logically deleted and nondeleted messages.

Once a queue maintenance task is in progress, the task flags the Queue Information entry and logically prevents any other task from accessing this queue. Any attempt to open this queue is rejected with the following message:

Queue has xxxx tasks attached. These must be purged.

The only action available at this point is to wait and try again later.

Monitoring functions

Selecting option 3 (Monitoring) from the master terminal main menu, (see Figure 13 on page 43) displays the screen shown in Figure 34:

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
11:58:29      *** Monitor Main Menu ***                    VSE1
MQMMMON                                               A005

                SYSTEM IS ACTIVE

                1. Monitor Queue
                2. Monitor Channel

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process      PF2 = Main Menu      PF3 = Quit

```

Figure 34. Monitor Main Menu

Monitor queues

Selecting 1 on the Monitor Main Menu allows you to monitor the current status of all existing local queues, using the screen shown in Figure 35:

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
09:03:11      Monitor Queues                                VSE1
MQMMMQQ                                               A005

                QUEUING SYSTEM IS ACTIVE

QUEUE          FILE      T INBOUND  OUTBOUND  LR      QDepth
ANYQ           MQFI002  N STOPPED STOPPED   0       0
BRET.LQ1       MQFI002  N IDLE    IDLE     17      0
BRET.MVS.XQ1   MQFI002  Y IDLE    IDLE     0       0
BRET.OS2.XQ1   MQFI002  Y IDLE    IDLE     0       0
BRET.XQ1       MQFI002  Y IDLE    IDLE     0       0
POP.BOTTLE     MQFI002  N IDLE    IDLE     0       0
QL.VSEP        MQFI002  N IDLE    IDLE     0       19
SYSTEM.EXCEPT MQFI001  N IDLE    IDLE     0       0
SYSTEM.LOG     MQF0001  N IDLE    IDLE     0       433
SYSTEM.MONITOR MQF0001  N IDLE    IDLE     0       4
..More

Information displayed.
ENTER = Refresh      PF2 = Main Monitor
PF7 = Back   PF8 = Forward   PF9 = All   PF10 = Detail

```

Figure 35. Monitor queues

Monitoring functions

This screen displays the current status of all local queues. The displayed fields are:

Queue

Name of the queue.

File

CICS FCT DDNAME of a local queue definition.

T

Queue type.

N Normal local queue.

Y Transmission local queue.

When PF9 (All) option is selected, additional type values may be displayed.

They are:

M Manager alias.

A Queue alias.

X Remote queue definition.

Inbound

Status of the inbound process.

ACTIVE One or more users have the queue open for MQPUT operations.

IDLE No user has the queue open for MQPUT operations.

STOPPED Queue has been stopped.

MAX At maximum queue depth.

FULL No space.

RECOVERY For dual queuing.

Outbound

Status of the outbound process.

ACTIVE One or more users have the queue open for MQGET operations.

IDLE No user has the queue open for MQGET operations.

STOPPED Queue has been stopped.

RECOVERY For dual queuing.

LR

Last Read: relative record number of the last record on the queue that has been read and processed.

Note: MQSeries messages are logically, rather than physically, deleted from the queue file. **LR** tells you which physical record is prior to the first active record.

QDepth

Estimated queue depth: The approximate number of records currently on the queue, remaining to be processed.

Notes:

1. The estimated queue depth is based on all MQPUT requests and on syncpointed MQGET requests.
2. Pressing the PF9 (All) function key displays an entire list of all queues (local, remote, and alias) together with their associated reference.
If you press this key again, the display lists local queues only.
3. Pressing the PF10 (Detail) function key displays detailed information for this local queue entry.

Monitor queues - detail

Pressing PF10 (Detail) displays detailed information for a specific queue on the screen shown in Figure 36:

```

07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
12:07:05           Monitor Queues                                VSE2
MQMMMMQ                                                    0002

                      QUEUING SYSTEM IS ACTIVE

                      DETAIL QUEUE INFORMATION

MVS1_TQ
INBOUND:  STATUS I  ENABLED Y  OPEN Q      0
          CHECKPOINT:          TAKEN      0THRESHOLD    1000
OUTBOUND:  STATUS I  ENABLED Y  OPEN Q      0
          CHECKPOINT:          TAKEN      0THRESHOLD    1000

BOTH:     FIQ      0  LIQ      4  GETS      0  QDEPTH      4
TRIGGER:  MAX      1  USED      0  TRAN      PROG. MQPSEND

CID VSE2_TO_MVS1

Information displayed.
ENTER = Refresh      PF2 = Main Monitor
                                      PF10 = List
    
```

Figure 36. Monitor Queues - detail

Monitor channel

Selecting 2 on the Monitor Main Menu (Shown in Figure 34 on page 75) allows you to monitor the current status of existing local communications channels, using the screen shown in Figure 37:

```
07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
09:28:33        Monitor Channels                      VSE1
MQMMMOOC                                               A005

                CHANNEL SYSTEM IS ACTIVE

CHANNEL        TYPE      MSN      QSN      QUEUE
BRET.CLIENT.TO.VSE  RECV      0        0 I
BRET.OS2.TO.VSE    RECV      98       10 I BRET.LQ1
BRET.VSE.TO.MVS    SEND       1        1 I BRET.MVS.XQ1
BRET.VSE.TO.OS2    SEND      54        2 I BRET.XQ1
SD01_LU62_VSEP     RECV      98        0 I
SD01_TCP_VSEP      RECV       2       23 I QL.VSEP
VSEP_LU62_SD01     SEND       0        0 I

Information displayed.
  ENTER = Refresh      PF2 = Main Monitor
  PF7 = Scroll Back   PF8 = Scroll Forward   PF10 = Detail
```

Figure 37. Monitor channel definitions

This screen displays the current status of local channels.

Channel

Name of the channel.

Type

Sender (SEND) or receiver (RECV).

MSN

Last channel message sequence number received or sent.

QSN

Queue message sequence number, of the queue name displayed in the **Queue** field.

QUEUE

Name of the queue associated with the channel. This field is preceded by a single character identifying the channel status:

I IDLE

B BUSY

C CLOSED (for example, DISABLED)

Note: If this is a receiver channel, the QUEUE field displays the last queue name for which a message has been received.

Monitor channel - detail

Pressing PF10 (Detail) displays detailed information for a specific channel shown in Figure 38:

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
12:10:39        Monitor Channels                          VSE2
MQMMOC                                                0002

                CHANNEL SYSTEM IS ACTIVE

                DETAIL CHANNEL INFORMATION

VSE1_TO_VSE2
RECEIVER  COMMIT  MSN    QSN DATE/TIME
          BEFORE   0      0 19970128102830
          AFTER   9      4 19970128102830
          QL.DEVL.X

Information displayed.
ENTER = Refresh      PF2 = Main Monitor
                                      PF10 = List

```

Figure 38. Monitor channel definitions - detail

This screen displays the channel name, the channel type, and the name of the queue it accesses. The MSN, QSN and time stamp of the last commitment for the BEFORE and AFTER COMMIT fields are also shown.

Browse function

Selecting option 4 (Browse Queue Records) from the master terminal main menu, (see Figure 13 on page 43) displays the screen shown in Figure 39:

```

07/16/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
09:32:14        Browse Queue Records                      VSE1
MQMDISP        SYSTEM IS ACTIVE                          A005

  Object Name: SYSTEM.LOG
  QSN Number : 00000005  LR-      0, LW-      433, DD-MQF0001
                    Queue Data Record
Record Status : Written.      PUT date/time : 19980702143235
Record Size   : 00000711     GET date/time :
Queue line.
MQI005000 PRG:MQPRECV TRN:MQ01 TRM:-997 TSK:00096 07/02/1998 14:32:35
CHANNEL CONNECTED

CHANNEL ID:     SD01_LU62_VSEP

EIBFN:  0C04      EIBRCODE: 000000000000 EXEC LINE: 000000
EIBRESP: 00000000 EIBRESP2: 00000000  EIBRSRCE:      ABCODE:

Information displayed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process  PF2 = Main Menu  PF3 = Quit  PF4 = Next  PF5 = Prior
PF7 = Up        PF8 = Down      PF9=Hex/Char  PF10=Txt/Head  PF12=Info/Sw

```

Figure 39. Browse Queue Records

This screen shows the content of the message for the specified queue sequence number (QSN) of the chosen object name (queue name). Record status is shown as Written or Deleted along with the associated time stamps.

To browse the queue records, enter the local object name and the QSN of the message of interest. The message on the queue appears in the blank area of the screen, and the message can be manipulated using the function (PF) keys.

Pressing the PF9 (Hex/Char) key displays the message in hexadecimal or EBCDIC text code.

The PF10 (Txt/Head) key presents detailed MQSeries information for the selected record, and includes channel information if the queue is a transmission queue.

If you browse the System Log file, the PF12 (Help) key appears and can be used to display user action and system action for this message, provided that the system is active.

Notes:

1. If the file you are browsing is in the process of being updated by any other MQSeries tasks, this function waits until the completion of those tasks. There can be a delay in the response of the browse function.
2. The Browse utility also has the ability to browse from a queue when the queue manager is not active.

Communications process

MQSeries uses the message channel agent (MCA) and the MQSeries listener for its communications.

The MCA process:

- Runs as a separate CICS task connected to the remote MQSeries using APPC or TCP/IP protocol.
- Starts automatically in response to other system activity, or when a message is placed on a transmission queue.
- Starts the MQSeries server when initial client requests are received.
- Can be stopped from the Operations Main Menu.

Select 2 from the Operations Main Menu to control the channels. See “Open / Close channel” on page 69 for further information.

The MQSeries listener process:

- Establishes itself as a TCP/IP “listener” process by binding itself to a port number configured in the global system definition.
- Runs as a separate CICS task waiting for remote TCP/IP connection requests.
- Starts the receiver MCA when connection requests are received.
- Ends when MQSeries is shut down.

Viewing error logs

MQSeries error messages, and other system messages, are placed on the following queues:

SYSTEM.LOG

All MQSeries generated error messages are written to this queue. If SYSTEM.LOG is not defined, or if MQSeries cannot successfully write to it, the error messages are logged to CSMT and may be viewed using standard system utilities.¹

SYSTEM.EXCEPT

Is the MQSeries dead letter queue. Messages that cannot be queued to their specified destination are queued here.

SYSTEM.MONITOR

API monitor queue used to log all application requests and their results. This is primarily for problem determination purposes.

Notes:

1. The names listed for these queues are the default names, but you can redefine the actual queue names through the global system definition screen.
2. You can view the messages written to these queues using the MQSeries browse queue function (see “Browse function” on page 80).

¹ This can be changed in the global system definition

Error logs

Chapter 5. MQSeries for VSE/ESA utility functions

MQSeries for VSE/ESA is supplied with various utility functions, which are:

- The MQSeries System Administration Control Interface – see “System Administration Control Interface”
- Batch modules – see “Background batch modules” on page 86
- The batch interface – see “Using the batch interface” on page 88
- Utilities for reclaiming VSAM file space – see “VSAM file maintenance” on page 90

System Administration Control Interface

The MQSeries System Administration Control Interface allows a limited number of system administration functions to be performed using programs.

The System Administration Control Interface has:

- A transactional interface (MQCL)
- A programmable interface (MQPCMD)

Transactional interface (MQCL)

MQCL is the command-line interface to the MQPCMD program. It allows you selectively to open and close:

- Queues
- Inbound channels
- Outbound channels
- Channels in both directions

The format of the command is:

```
MQCL FF NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
```

Where:

MQCL Command line transaction ID.
FF Function code. Valid codes are:
 CR Close receiver channel.
 IDOR Open receiver channel ID.
 CS Close sender channel.
 IDOS Open sender channel ID.
 SB Start queue both directions.
 SI Start queue inbound direction.
 SO Start queue outbound direction.
 XI Stop queue both directions.
 XI Stop queue inbound direction.
 XO Stop queue outbound direction.
NNNN The name of a queue or channel.

A message is sent to the activated terminal and the SYSTEM.LOG for every activation of this task. This message has an error code of MQM001000 for

Administration interface

completed messages, or an error code of MQM001090 for any that did not complete properly. The text that follows explains the exact results.

Note that error code MQM001090 indicates invalid syntax in the command line program, MQCL.

Programmable interface (MQPCMD)

The programmable interface uses an EXEC CICS LINK PROGRAM (MQPCMD). MQPCMD accepts a fixed format COMMAREA which specifies the type of request and a status response area.

The supplied copybook MQICMD.C describes this area.

```
*-----*
* - BEGIN -      *** COPYBOOK: MQICMD      ***      - BEGIN - *
*-----*
* COMMAND LINE COPYBOOK *
*-----*

01 MQI-COMMAND-LINE.
02 MQI-CMD-PASSED-AREA.
05 MQI-CMD-TRANS-ID      PIC X(4) VALUE 'MQCL'.
05 FILLER                PIC X VALUE SPACE.
05 MQI-CMD-FUNCTION      PIC XX VALUE SPACE.
    88 MQI-CMD-FUNCTION-OK VALUE 'CR', 'OR',
                                'CS', 'OS',
                                'XB', 'XI', 'XO',
                                'SB', 'SI', 'SO'.
    88 MQI-CMD-FUNC-STATION VALUE 'CR', 'OR',
                                'CS', 'OS'.
    88 MQI-CMD-CLOSE-STATION VALUE 'CR', 'CS'.
    88 MQI-CMD-OPEN-STATION VALUE 'OR', 'OS'.
    88 MQI-CMD-STATION-SEND VALUE 'CS', 'OS'.
    88 MQI-CMD-STATION-RECVR VALUE 'CR', 'OR'.

    88 MQI-CMD-FUNC-QUEUE   VALUE 'XB', 'XI', 'XO',
                                'SB', 'SI', 'SO'.
    88 MQI-CMD-STOP-QUEUE  VALUE 'XB', 'XI', 'XO'.
    88 MQI-CMD-STOP-Q-INBOUND VALUE 'XI'.
    88 MQI-CMD-STOP-Q-OUTBOUND VALUE 'XO'.
    88 MQI-CMD-STOP-Q-BOTH  VALUE 'XB'.

    88 MQI-CMD-START-QUEUE  VALUE 'SB', 'SI', 'SO'.
    88 MQI-CMD-START-Q-INBOUND VALUE 'SI'.
    88 MQI-CMD-START-Q-OUTBOUND VALUE 'SO'.
    88 MQI-CMD-START-Q-BOTH  VALUE 'SB'.

    88 MQI-CMD-Q-IN        VALUE 'SI', 'XI'.
    88 MQI-CMD-Q-OUT       VALUE 'SO', 'XO'.
    88 MQI-CMD-Q-BOTH      VALUE 'SB', 'XB'.

05 FILLER                PIC X VALUE SPACE.
05 MQI-CMD-QUEUE-NAME     PIC X(48) VALUE SPACES.
05 MQI-CMD-STATION-NAME   REDEFINES
    MQI-CMD-QUEUE-NAME     PIC X(48).

*--RETURNED INFO ONLY ON CICS LINK
02 MQI-CMD-RETURNED-AREA.
05 MQI-CMD-RC             PIC S9(4) COMP VALUE ZERO.
05 MQI-CMD-RC-OK         VALUE ZERO.
```



```

      88 MQI-CMD-RC-DUPLICATE-FUNC          VALUE +4.
      88 MQI-CMD-RC-NAME-INVALID          VALUE +10.
      88 MQI-CMD-RC-SYS-NOT-ACTIVE        VALUE +80.
      88 MQI-CMD-RC-ERRORS                VALUE +90.
      88 MQI-CMD-RC-HELP                  VALUE +99.

05 MQI-CMD-ERROR-LINE.
   10 MQI-CMD-ERROR-PREFIX                PIC XXX  VALUE 'MQM'.
   10 MQI-CMD-ERROR-CODE                 PIC X(6) VALUE SPACES.
   10 FILLER                              PIC X   VALUE SPACE.
   10 MQI-CMD-ERROR-TEXT                 PIC X(40) VALUE SPACES.
   10 FILLER                              PIC X   VALUE SPACE.
   10 MQI-CMD-ERROR-NAME                 PIC X(48) VALUE SPACES.

*-----*
* - END -          *** COPYBOOK: MQICMD          ***          - END - *
*-----*

```

The following sample uses MQICMD.C:

```

IDENTIFICATION DIVISION.
PROGRAM-ID.    TESTCMD.
AUTHOR.       IBM.

DATE-COMPILED.

*
* SUMMARY CHANGES:
*-----*
*
* CHANGE  DATE      PGM  COMMENT
*
*
*-----*
*****
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
DATA DIVISION.

WORKING-STORAGE SECTION.

*-----*
01 WS-QCLOSED      PIC X(20) VALUE 'Queue closed OK.'.
01 WS-QNCLOSED    PIC X(20) VALUE 'Queue not closed.'.
      COPY MQICMD.
*-----*
LINKAGE SECTION.
*-----*
*-----*
PROCEDURE DIVISION.
*-----*

0000-MAIN-LINE.
      MOVE 'QL.VSEP' TO MQI-CMD-QUEUE-NAME
      SET MQI-CMD-STOP-Q-BOTH TO TRUE
      EXEC CICS LINK
              PROGRAM('MQPCMD')
              COMMAREA(MQI-COMMAND-LINE)
              LENGTH(LENGTH OF MQI-COMMAND-LINE)
      END-EXEC
      EVALUATE TRUE
      WHEN MQI-CMD-RC-OK
      EXEC CICS WRITE OPERATOR
              TEXT(WS-QCLOSED)

```

Batch modules

```
                                TEXTLENGTH(LENGTH OF WS-QCLOSED)
                                END-EXEC
                                WHEN OTHER
                                EXEC CICS WRITE OPERATOR
                                TEXT(WS-QNCLOSED)
                                TEXTLENGTH(LENGTH OF WS-QNCLOSED)
                                END-EXEC
                                END-EVALUATE
                                .
*-----*
0000-RETURN.
EXEC CICS RETURN
END-EXEC
.
GOBACK.
```

Background batch modules

The PRD2.MQSERIES library contains all the sample code and JCL, including the MQJUTILY.Z example background batch job.

MQJUTILY.Z contains the MQPUTIL program, which performs the following functions:

- Prints the system, queue, and channel definitions from a configuration file.
- Prints the SYSTEM.LOG file in a formatted report.
- Updates all channels with a new starting MSN.
- Updates a configuration file for dual queues. It makes all dual queues into a primary queue.
- Prints new Help Facility error information.

The MQPUTIL program uses the CONFIG DLBL for the MQSeries configuration VSAM file, if the PRINT LOG command is used. The MQPUTIL program uses the following general syntax:

Column	Content
1 to 5	Command name
6	Space
7 to 18	Subcommand
19	Space
20..	Arguments

MQPUTIL program

The supplied MQPUTIL program has three commands, which are:

- PRINT
- RESET
- DUALQ

Note: You must run the MQPUTIL program once for each command that you require.

PRINT

PRINT has three options:

CONFIG

Prints the full MQSeries configuration using the sample DLBL provided.

LOG

Prints the system log in a formatted report using the sample DLBL provided.

MESSAGES

Prints a Help Facility resolution report.

RESET

RESET has two options:

MSN nnnnnnnn

Resets all channel numbers to nnnnnnnn

CHECKPOINT

Resets all channel checkpoint values to zero.

RESET CHECKPOINT causes the channel records to be updated with a new checkpoint value. This causes the current MSN values to be maintained when MQSeries is started.

MQSeries compares the latest checkpoint value for a channel against that of the updated channel date and time for a queue record. If the queue record has a higher value, the MSN in the queue record is used. Therefore, whichever checkpoint value was taken last becomes the current MSN when MQSeries is started.

If the MQPUTIL program is used to perform the RESET CHECKPOINT function, no queue scan is performed.

DUALQ

DUALQ has the following option:

TAKEOVER dual_queue_name

Allows the dual queue specified to become the primary queue, using the following process:

1. The configuration file points to the cluster hosting the dual queue instead of to the cluster hosting the primary queue.
2. All message headers in the dual queue are modified to contain the name of the primary queue instead of the name of the dual queue.

This command may be used when a local queue becomes unavailable, for example, when input or output errors occur, and a dual queue has been defined.

Note: You are recommended to backup the configuration file, using the VSAM REPRO command, before using this command, because the file will be changed. The configuration file can be restored when you have repaired the failure.

Using the batch interface

MQSeries for VSE/ESA has been designed for on line programs only. However, in a few cases, it can be worth using batch programs as well. For this purpose, four sample programs are provided:

MQBIBTCH.Z
Batch interface assembler program.
MQBICALL.Z
COBOL sample application.
MQBICIRH.Z
CICS COBOL request handler.
MQBICITK.Z
CICS interface assembler.

Note: These programs are supplied in a basic format and are provided on an “as-is” basis. You may have to modify them to fit your environment.

Batch interface logic

Only CICS programs can issue MQSeries requests. Therefore, the batch program runs a CICS transaction that actually issues the MQSeries requests. Data transportation is performed using the XPCC SEND and REPLY protocol.

To handle all XPCC requests a VSE subtask is attached to the CICS partition.

. Two assembler programs issue XPCC requests:

- MQBIBTCH on the batch side
- MQBICITK on the CICS side

The programs work as follows, with MQSeries feedback following the reverse path.

```
program  ----- Call -----> MQBIBTCH -----> XPCC -----> MQBICITK -----> LINK -----> MQBICIRH -----> Call -----> MQSeries
```

Using the Batch interface

1. Issue MQSeries functions in your batch program, just as you do in CICS programs. For example:

```
CALL 'MQCONN' USING  
    QM-NAME-AREA  
    HCONN-ADDR-AREA  
    CCODE-ADDR-AREA  
    RCODE-ADDR-AREA.
```

2. Link-edit your program by including module MQBIBTCH. For example:

```
// JOB GEGETST  
// OPTION CATAL  
PHASE MYPROG,*  
// EXEC IGYCRCTL,....  
    your program here  
/*  
INCLUDE MQBIBTCH  
// EXEC LNKEDT  
/&
```

3. Start the CICS interface, using the transaction MQBI.

4. Run your batch program.

Note: When your program has ended, the CICS counterpart is still activated. Therefore, you can start another batch program without restarting the transaction MQBI. To deactivate the batch interface, the batch program must issue a CALL 'MQBIEND'.

For example, this small batch program stops the batch interface:

```
ID DIVISION.
PROGRAM-ID.  MQBISTOP.
AUTHOR.     IBM.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
PROCEDURE DIVISION.
CALL 'MQBIEND'.
GOBACK.
```

Data integrity

To test for data integrity the following functions are used:

- MQCMIT commits all changes. This forces a CICS SYNCPOINT to be issued by the mirror transaction.
- MQBACK rolls back all changes. The CICS mirror transaction issues EXEC CICS SYNCPOINT ROLLBACK.

For both functions the syntax is as follows:

```
CALL 'funct' USING
      HCONN-ADDR-AREA
      CCODE-ADDR-AREA
      RCODE-ADDR-AREA.
```

Notes:

1. None of the passed parameters is actually tested or used.
2. Under CICS, updates are not automatically committed. However, if a batch program issues the MQDISC call while there are uncommitted requests, an implicit syncpoint occurs.
3. An implicit rollback occurs, and the batch interface VSE subtask (under CICS) is ended if one of the following happens:
 - a. An MQBIEND call is issued without a previous MQDISC call.
 - b. A system error condition is detected. For example, the batch program ends without issuing an MQDISC call.

Verifying the batch interface

The batch program MQBICALL has been provided for this purpose. You can use the following job as a test:

VSAM file maintenance

```
// JOB CALLER
// LIBDEF *,SEARCH=(PRD2.MQM SERIES,PRD2.SCEEBASE)
* Put 5 messages into queue: GEGE
// EXEC MQBICALL
PUT 005 GEGE
/*
/. END
/&
```

Restrictions on using the batch interface.

1. Only one batch program can be running at a time against an MQSeries queue manager.
2. Only one CICS partition can run the batch interface at a time.

However, by changing the application names in XPPC IDENT, you may have multiple versions running. However, only one batch program can communicate with one CICS partition.

3. The MQINQ function has the following limitations:
 - A maximum of 10 selectors
 - A maximum of 10 integer attributes
 - 500 characters for the character attribute buffer

Otherwise, you need to modify MQBITCH and QMBICIRH.

VSAM file maintenance

All files used by MQSeries are VSAM clusters. Most of these contain queues and need to be reorganized from time to time.

A queue is an ordered suite of VSAM records in a KSDS organization. Each record key is 56 bytes long, 48 being for the queue name, and eight for the Queue Sequence Number (QSN) and other information. This QSN is assigned sequentially, resulting in all keys being created in ascending order.

Even when a queue record is physically deleted from a queue, the space it occupies is not reclaimed due to the way VSAM works. Therefore, unless you reclaim the space used by these records, there is the possibility that you will obtain a VSAM “space full” condition.

The queue dump facility allows you to rebuild an MQSeries VSAM queue file. This eliminates processed messages and fully regains VSAM free space.

There are two ways of reclaiming the space of deleted records:

1. Using the on line “Delete All” function through the MQMT screens. See “Delete all function” on page 91 for more information.
2. Using the MQPREORG batch program. See “MQPREORG function” on page 91 for more information.

Delete all function

On the Maintain Queue Records screen (see “Queue maintenance” on page 73), there is a function called “Delete All”. This function physically deletes all messages, and resets the QSN to one, in order to reclaim freed space.

This is a useful tool to maintain the system log file for MQSeries. The advantage of this function is that it is an on line function requiring no other manual operation.

Attention: Note that this function deletes all messages and should not be used on queue files that contain undelivered messages.

Operation

1. On the Start/Stop Queue Control screen, stop the desired queue; see Figure 29 on page 67.
2. If the desired queue is a transmission queue, stop only the inbound direction first. When the queue depth reaches zero, stop the outbound direction and close the associated sender channel.
3. If the desired queue is a destination queue with trigger capability, close the associated receiver channel.
4. On the Maintain Queue Records screen enter the queue name, together with a function of A, and press the PF6 (Update) function key; see Figure 33 on page 73.
5. Press the Enter key to display the result.
6. After “Queue Processing Finished” is displayed, start the reorganized queue on the Start/Stop Queue control screen.

MQPREORG function

MQSeries includes a batch program utility called MQPREORG, and sample JCL to run MQPREORG.

This utility can be used as a nightly, or weekly, queue maintenance facility on any number of queue files. You can also specify a date and time to carry out the procedure. The utility accepts the queue name from SYSIPT and the name of the VSAM file from DLBL.

All messages are ignored, except those marked as “Written” (to be delivered after a specified date and time) on the specified queue. The retained messages are resequenced and placed in a work file.

After the VSAM cluster is deleted and redefined, the retained and resequenced messages are copied back into it. If none of the written messages is to be retained, you can use a “delete-and-define” IDCAMS JCL to do the job.

Multiple queues sharing a VSAM cluster

Attention: Although it is possible for MQSeries for VSE/ESA queues to share the same VSAM file cluster, this is **not** advised. To give maximum independence to data, each queue should be assigned a unique VSAM file cluster.

VSAM file maintenance

This is particularly important if the queue is defined for automatic reorganization. See “Creating local queues” on page 49 and “Automatic reorganization” on page 53.

If there is more than one queue defined in a VSAM cluster, all queues have to be processed before deleting and recreating this cluster. Otherwise, records from unprocessed queues will be lost.

To help you reorganize all queues, you may use the “ALL” option instead of the queue name, as follows:

```
// EXEC MQPREORG
ALL
/*
```

To reorganize a specific queue, enter one of the following commands:

```
// EXEC MQPREORG
LQ.INVOICE
/*
```

or

```
// EXEC MQPREORG
LQ.INVOICE YYYYNNDDHHMMSS
/*
```

where:

YYYY	Is the year
NN	Is the month
DD	Is the day
HH	Is the hour
MM	Is the minute
SS	Is the second

Reorganizing queue files

This procedure is to be used only when the queue manager is not running.

1. If CICS is running, use CEMT to shut down and close the VSAM files you are going to process.
2. Modify the sample JCL to include your system parameters and reorganization requirements.
3. Process the job to run the batch program utility, MQPREORG, to reorganize the VSAM files and reclaim all freed space.
4. If you performed step 1, use CEMT to open and enable the processed VSAM files.

Sample JCL to run MQPREORG

```

* ** JOB JNM=MQJREORG,DISP=D,CLASS=0
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJREORG - Re-Organize MQ/Series for VSE/ESA queues.
* -----*
*   I M P O R T A N T   I M P O R T A N T   I M P O R T A N T   *
*   *
*   Please change : *
*       "* ** JOB" to "* $$ JOB" *
*       "* ** LST" to "* $$ LST" *
*       "* ** EOJ" to "* $$ EOJ" *
*   *
*   Fields filed with ?valid? have also to be modified to suit the *
*   user specifications. *
*   -----*
*   *
*   This job deletes delivered messages from an MQSeries queue in *
*   order to reclaim the DASD freed space. *
*   *
*   INPUT to MQPREORG : *
*   (only one statement is allowed, delimited by one or more spaces)*
*   *
*   1. Any QUEUE name delimited by one or more spaces *
*   (In this JCL, only queue OS2_LOCAL is to be processed) *
*   If there are any other queues in the same cluster, *
*   they will be echoed into OUTPUTQ. *
*   2. If you want to process EVERY queue in a cluster, *
*   please key in "ALL ". *
*   *
*   This sample assumes we want to reorganize queues defined to the *
*   VSAM cluster MQIF002. Changes must be made for other clusters. *
*   -----*
*   Licensed Materials - Property of IBM *
*   *
*   5686-A06 *
*   (C) Copyright IBM Corp. 1998 *
*   *
*   US Government Users Restricted Rights - Use, duplication or *
*   disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
*   -----*
// DLBL INPUTQ,'MQSERIES.MQFI002',,VSAM,CAT=MQMCAT
// DLBL OUTPUTQ,'MQSERIES.WORK.QUEUE',,VSAM,CAT=MQMCAT
// EXEC IDCAMS,SIZE=AUTO
/* *
/*           VERIFY VSAM FILE *
/* *
        VERIFY FILE(INPUTQ)
        IF MAXCC > 0 THEN CANCEL /* This means Cluster in use */

        DELETE (MQSERIES.WORK.QUEUE) -
                CL ERASE PURGE CAT(?CAT?)
        SET MAXCC = 0
        DEFINE CLUSTER -
                (NAME (MQSERIES.WORK.QUEUE) -
                CYLINDERS (10 10) -
                VOLUMES (?valid?) -
                NONINDEXED) -
        DATA -
                (NAME (MQSERIES.WORK.QUEUE.DATA) -
                RECORDSIZE (2048 32048) -
                CISZ (8096)) -
                CAT (?CAT?)
/*

```

VSAM file maintenance

```
// IF $MRC GT 0 THEN
// GOTO WRAPUP
// LIBDEF PHASE,SEARCH=(PRD2.MQSERIES,PRD2.SCEEBASE)
// EXEC MQPREORG,SIZE=AUTO
OS2_LOCAL
/*
// IF $MRC GT 0 THEN
// GOTO WRAPUP
// EXEC IDCAMS,SIZE=AUTO
        DELETE      (MQSERIES.MQFI002)                -
                CLUSTER NOERASE PURGE CATALOG (?CAT?)
SET MAXCC = 0
/*                                                    */
        DEF CLUSTER(NAME(MQSERIES.MQFI002)            -
        FILE(MQFI002)                                -
        VOL(?volid?)                                  -
        RECORDS (3000 100)                            -
        RECORDSIZE (200 4089)                         -
        INDEXED                                        -
        KEYS(52 0)                                     -
        SHR(2))                                         -
        DATA (NAME (MQSERIES.MQFI002.DATA) CISZ(4096)) -
        INDEX (NAME (MQSERIES.MQFI002.INDEX) CISZ(1024)) -
        CATALOG(?CAT?)
        IF LASTCC > 0 THEN CANCEL
/*                                                    */
/*          Execute REPRO only of the define  was OK.    */
/*                                                    */
REPRO INFILE(OUTPUTQ) OUTFILE(INPUTQ)
IF LASTCC > 0 THEN CANCEL
/*                                                    */
/*          Delete only if REPRO was OK.                */
/*                                                    */
/*                                                    */
DELETE      (MQSERIES.WORK.QUEUE)                    -
                CL ERASE PURGE CAT(?CAT?)
/*
/. WRAPUP
/&
* ** EOJ
```

Chapter 6. Problem determination

This chapter suggests reasons for some of the problems you may have using MQSeries for VSE/ESA. The process of problem determination is that you start with the symptoms and trace them back to their cause.

Not all problems can be solved immediately, for example, performance problems caused by the limitations of your hardware. Also, if you think that the cause of the problem is in the MQSeries code, contact your IBM Support Center.

The cause of your problem could be in:

- MQSeries setup and local queue operation
- The network
- The application
- Other areas of investigation

The sections that follow raise some fundamental questions that you need to consider. Work through the questions, making a note of anything that might be relevant to the problem.

MQSeries setup and local queue operation

You should ensure that MQSeries for VSE/ESA is installed correctly and working with local queues before you investigate any other problems.

Has MQSeries run successfully before?

If MQSeries has not run successfully before, it is likely that you have not yet set it up correctly. See “MQSeries installation verification test” on page 19 to check that you have carried out all the steps correctly, and set up a SYSTEM.LOG queue as follows:

1. Define a queue name as SYSTEM.LOG using:
 - a. A physical file name MQFLOG using the file name from the file control table
 - b. A maximum queue depth of 1 000 000

See step 3d on page 17 and step 3f on page 17 in “MQSeries installation verification test” on page 19 for information about defining a queue.

2. Enter SYSTEM.LOG as the object name with a valid QSN number.

You can browse the log queue by selecting 4, Browse Queue Records, on the Master Terminal Main Menu, as described in “Browse function” on page 80.

See “Global system definition” on page 44 for more information.

Is local queue operation working?

This may require the creation of a local queue definition as described in “MQSeries initialization” on page 16. In addition, check that the VSAM files referenced in the queue definition are open and correctly enabled.

Use:

```
CEMT INQUIRE FILE (filename)
```

to ensure that the VSAM file associated with the queue is accessible.

Use the instructions described in “Local queue verification test” on page 19 to test a local queue. Ensure that you can:

- Put and get messages to the local queue, using the supplied test transaction TST2.
- Browse the queue correctly using the MQMT System Administration Browse function.

Network problems

Before MQSeries for VSE/ESA can use an inbound or outbound channel connection to an SNA-connected MQSeries platform, a connection must already be established between CICS/VSE and the remote platform.

The person responsible for the VTAM and CICS definitions in your enterprise should perform the following investigations.

Investigating SNA problems

If an attempt to start a channel fails, it may be the result of a session failure. If it is not possible to establish a session between CICS and the LU for the remote channel endpoint, either before or during the channel attempting to start, the connection fails.

Enter the following command if you suspect that a session failure is causing the problem:

```
D NET,ID=<remote lu name>,E
```

This gives details of the LU which should be in session with CICS, and also lists any sessions it currently has.

Notes:

1. Look at the session limit for the LU. If it is shown as one for an independent LU, there is a problem with the SNA definitions.
2. See if <minor node name> is listed amongst the sessions. If it is, there is a session between the LU and CICS. This indicates that the problem may not be at the network level, or that there are insufficient sessions between the two LUs to support a new channel request.

Enter the command again, to see whether for this session, the send and receive counts have changed, indicating the session is in use.

If the command returns “PARAMETER VALUE INVALID”, this means that VTAM does not know the <remote lu name>. Either you entered the name incorrectly, or VTAM cannot locate it. Try defining the name again and attempt to start the channel.

If VTAM is able to display <remote lu name>, try the following command in CICS:

```
CEMT I CONN(<remote conn>)
```

This shows the status of the connection from CICS to the remote system. Next to the entry is an indication showing it to be INService or OUTservice and ACQuired or RELeased. The status needs to be Inservice and Acquired.

```
CEMT I MODE CONN(<remote conn>)
```

This command displays the status of the mode names associated with the connection. For connections supporting parallel sessions, there will be at least two mode names, SNASVCMG and <logmode 1>, showing the number of active sessions for each.

If the SNASVCMG group has no sessions active, the connection is in a RELeased state, rather than an ACQuired state.

These sessions are SNA services manager sessions, and not used by MQSeries channels. However, at least one of the two needs to be active for the connection to be usable.

If the remote LU has been incorrectly defined, so that it has a session limit of one, it is possible that one SNASVSMG session is active, but that no other sessions can be established, including those required by the MQSeries channel.

The <logmode 1> sessions may be used by MQSeries channels.

For single session connections, one mode name, <logmode 2>, is shown with just one session in the group.

The MQSeries channel must have been set up to use the logon mode <logmode 1>, or <logmode 2>, as appropriate.

Investigating TCP/IP problems

Is TCP/IP able dynamically to establish a session between nodes in the network? Use the following instruction to test a connection to a remote TCP/IP node:

```
[ping.nodename]
```

If you are unable to “ping” the remote TCP/IP node successfully, inform your VSE/ESA systems programmer who installed TCP/IP.

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 message queue manager is not working, the messages cannot flow to a remote queue.

Check that the connection between the two systems is available, and that the intercommunication component of MQSeries has been started.

Applications

Check that messages are reaching the transmission queue, and check the local queue definition of the transmission queue and any remote queues.

Have you made any network-related changes, or changed any MQSeries definitions, that might account for the problem?

Applications

The errors in the following list illustrate the most common causes of problems encountered while running MQSeries programs. You should consider the possibility that the problem with your MQSeries system could be caused by one or more of these errors:

- Assuming that queues can be shared, when they are in fact exclusive.
- Passing incorrect parameters in an MQI call.
- Passing insufficient parameters in an MQI call. This may mean that MQI cannot set up completion and reason codes for your application to process.
- Failing to check return codes from MQI requests.
- Passing variables with incorrect lengths specified.
- Passing parameters in the wrong order.
- Failing to initialize *MsgId* and *CorrelId* correctly.

Are there any error messages?

MQSeries uses the system log to capture messages concerning the operation of MQSeries itself, the queue manager, and error data coming from the channels that are in use. Check the system log to see if any messages have been recorded that are associated with your problem.

See “System log” on page 108 for information about the contents of the system log.

Are there any return codes explaining the problem?

If your application gets a return code indicating that a Message Queue Interface (MQI) call has failed, refer to the *MQSeries 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 it is reproduced:

- Is it caused by a command or an equivalent administration request?
Does the operation work if it is entered by another method? If the command works if it is entered on the command line, but not otherwise, check that the command server has not stopped.
- Is it caused by a program? Does it fail on all MQSeries systems and all queue managers, or only on some?
- 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 the MQSeries system, 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 are not aware of might have been run on the system.

- Have you changed, added, or deleted any queue definitions?
- Have you changed or added any channel definitions? Changes may have been made to either MQSeries channel definitions or any underlying communications definitions required by your application.
- Do your applications deal with return codes that they might get as a result of any changes you have made?

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. Is it possible to retry using a back level of the application?

- 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 the 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?

Has your MQSeries system been changed, perhaps in a minor way, such that 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?

- Does the application run on other MQSeries systems?

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

If the application has not run successfully before

If your application has not yet run successfully, you need to examine it carefully to see if you can find any errors.

Before you look at the code, and depending upon which programming language the code is written in, examine the output from the translator, or the compiler and linker, if applicable, to see if any errors have been reported.

If your application fails to translate, compile, or link, it will also fail to run if you attempt to invoke it.

If the documentation shows that each of these steps was accomplished without error, you should consider the coding logic of the application. Do the symptoms of the problem indicate the function that is failing and, therefore, the piece of code in error? See “Applications” on page 98 for some examples of common errors that cause problems with MQSeries applications.

Using the MQSeries API monitor

By selectively using the MQSeries API monitor, you can:

- Track precisely which MQSeries API issues an application
- Establish which return codes are passed

The MQSeries API monitor is started and stopped using the MQMT system administration transaction option 2.1, which is used to toggle the API monitor on and off.

```

08/12/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
10:31:09           Start / Stop Queue                             VSE1
MQMMS              A004

                System Information
System Status   : SYSTEM IS ACTIVE
Queue Status    : Queuing System is active.
Channel Status  : Channel System is active.
Monitor Status  : Monitor is not active.

                Single Queue Request
Queue Name      :
Function        :           S=Start, X=Stop, R=Refresh from Config
Mode           :           I=Inbound, O=Outbound, B=Both

INBOUND Status :
OUTBOUND Status :

                All Queues Request
Function        : M           S=Start, X=Stop, or M=Monitor

MONITOR STARTED.
  ENTER=Display    PF2 = Oper Menu    PF3 = Exit        PF6 = Update
  
```

Figure 40. API monitor

Once the API monitor is started, the application to be tested can be processed, and the API monitor stopped.

Note: The API monitor should be started for limited periods only. It traces the processing of all running applications, and consequently makes heavy usage of system resources.

Other areas

```
01 MQ-MONITOR-RECORD.
   05 MQ-MON-TRAN-ID      PIC X(4)  VALUE SPACES.
   05 MQ-MON-TERM-ID     PIC X(4)  VALUE SPACES.
   05 MQ-MON-CICS-TASKN  PIC S9(8) COMP VALUE ZERO.
   05 FILLER              PIC X(12) VALUE SPACES.

   05 MQ-MON-SYSTEM-NUM  PIC 99    VALUE 01.
   05 FILLER              PIC X(14) VALUE SPACES.
   05 MQ-MON-FUNCTION    PIC X(4)  VALUE SPACES.
   88 MQ-MON-CONNECT     VALUE 'CONN', 'CONI'
                           'MCCO'.
   88 MQ-MON-CONNECT-VIA-APPL VALUE 'CONN', 'CONI'.
   88 MQ-MON-CONNECT-VIA-INTERFACE VALUE 'CONI'.
   88 MQ-MON-MCP-CONNECT VALUE 'MCCO'.
   88 MQ-MON-OPEN        VALUE 'OPEN'.
   88 MQ-MON-PUT         VALUE 'PUT '.
   88 MQ-MON-INQ         VALUE 'INQ '.
   88 MQ-MON-GET         VALUE 'GET '.
   88 MQ-MON-CLOSE       VALUE 'CLOS'.
   88 MQ-MON-DISCONNECT VALUE 'DISC'.

   05 FILLER              PIC X(4)  VALUE SPACE.
   05 MQ-MON-START-ABSTIME PIC S9(15) COMP-3 VALUE ZERO.
   05 MQ-MON-END-ABSTIME  PIC S9(15) COMP-3 VALUE ZERO.
   05 MQ-MON-RESULTS.
   10 MQ-MON-CCODE        PIC S9(8)  COMP VALUE ZERO.
   10 MQ-MON-RCODE        PIC S9(8)  COMP VALUE ZERO.

   05 FILLER              PIC X(12) VALUE SPACES.
   05 MQ-MON-FUNCTION-DEP-INFO VALUE SPACES.
   10 MQ-MON-QM-NAME      PIC X(48).
   10 MQ-MON-Q-NAME       PIC X(48).
   10 MQ-MON-RESOLVED-Q  PIC X(48).
   10 FILLER              PIC X(200).
```

It is possible to follow the flow of MQSeries API calls using a specific application. The application is identified by its CICS transaction code, terminal identifier, and CICS task number.

The specific MQSeries API calls are identified, together with the queue manager name, queue name, and condition and return codes.

Ensure that you toggle the MQSeries API monitor **off** after use.

Other areas of investigation

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 MQSeries library (see “MQSeries publications” on page x) 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.

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

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 that best describes the symptoms of your problem.

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

Have you obtained incorrect output?

In this book, “incorrect output” refers to your application:

- Not receiving a message that it was expecting
- Receiving a message containing unexpected or corrupted information
- Receiving a message that it was not expecting, for example, one that was destined for a different application

In all cases, check that any queue or queue manager aliases that your applications are using are correctly specified and accommodate any changes that have been made to your network.

If an MQSeries error message is generated, all of which are prefixed with the letters “MQI,” you should look in the system log. See “System log” on page 108 for further information.

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, so these are the times when load-dependent problems are most likely to occur. (If your MQSeries network extends across more than one time zone, peak system loading might seem to occur at some other time of day.)

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 may issue an MQGET call, without specifying a wait option, before an earlier process has completed. An intermittent problem may also be seen if your application tries to get a message from a queue while the call that put the message is in-doubt (that is, before it has been committed or backed out).

Have you applied any service updates?

If a service update has been applied to MQSeries, check that the update action completed successfully and that no error message was produced.

- Did the update have any special instructions?
- Was any test run to verify that the update had been applied correctly and completely?
- Does the problem still exist if MQSeries is restored to the previous service level?
- If the installation was successful, check with the IBM Support Center for any patch error.

- If a patch has been applied to any other program, consider the effect it might have on the way MQSeries interfaces with it.

Does the problem affect only remote queues?

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

- Check that required channels have been started and are triggerable.
- Check that the programs that should be putting messages to the remote queues have not reported problems.
- If you use triggering to start the distributed queuing process, check that the transmission queue has triggering set on.
- Check the system log and VSE console for messages indicating channel errors or problems.

See the *MQSeries Intercommunication* book for information about how to define channels.

Is your application or MQSeries for VSE/ESA running slowly?

MQSeries for VSE/ESA runs as a subsystem, with a CICS partition, on the VSE operating system. The VSE operating system itself may be a second-level client on a VM machine. This complexity means that a performance problem can exist in any of these components.

MQSeries for VSE/ESA is sensitive to the CICS environment and availability of CICS resources. CICS performance problems are a specialized area requiring detailed analysis. Investigate these problems with the assistance of your CICS systems programmer.

MQSeries for VSE/ESA utilizes VSAM files under VSE. After prolonged use these files tend to fragment into several VSAM extents. This can be viewed with the VSE ICCF File and Catalog Management facility. Any files that show multiple extents should be reallocated with IDCAMS as soon as it is convenient to do so.

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

This could also be caused by a performance problem. Perhaps it is because your system is operating near the limits of its capacity.

Operating system performance problems, for both VSE/ESA and VM, are a specialized area requiring detailed analysis. Investigate these problems with the assistance of your VSE/ESA or VM systems programmer.

A performance problem may be caused by a limitation of your hardware.

If you find that performance degradation 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 occurs only when specific queues are accessed.

The following symptoms might indicate that MQSeries is running slowly:

- Your system is slow to respond to MQSeries commands.

- 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.

If the performance of your system is still degraded after reviewing the above possible causes, the problem may lie with MQSeries for VSE/ESA itself. If you suspect this, you need to contact your IBM Support Center for assistance.

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 MQSeries calls are discussed in the following sections.

For more information about application design, see the *MQSeries Application Programming Guide*.

Effect of message length

Although MQSeries allows messages to hold up to 4 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, you should send only the essential data in a message; for example, in a request to debit a bank account, the only information that may need to be passed from the client to the server application is the account number and the amount of the debit.

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.

Queues that contain messages of different lengths

If the messages on a queue are of different lengths, to determine the size of a message, your application could use the MQGET call with the *BufferLength* field set to zero so that, even though the call fails, it returns the size of the message data. The application could then repeat the call, specifying the identifier of the message it measured in its first call and a buffer of the correct size. However, if there are other applications serving the same queue, you might find that the performance of your application is reduced because its second MQGET call spends time searching for a message that another application has retrieved in the time between your two calls.

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 *MaxMsgLength* 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 4 MB, the maximum allowed by MQSeries for VSE/ESA.

Use 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.

Incorrect output

The term “incorrect output” can be interpreted in many different ways. For the purpose of problem determination within this book, the meaning is explained in “Have you obtained incorrect output?” on page 103.

Two types of incorrect output are discussed in this section:

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

Additional problems that you might find if your application includes the use of distributed queues are also discussed.

Messages that do not appear on the queue

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

- Has the message been put on the queue successfully?
 - Has the queue been defined correctly. For example, are the queue and maximum message length sufficiently large?
 - Is the queue enabled for putting?
 - Is the queue already full? This could mean that an application was unable to put the required message on the queue.
- Are you able to get any messages from the queue?

- Do you need to take a syncpoint?

If messages are being put or retrieved within syncpoint, they are not available to other tasks until the unit of recovery has been committed.

- Is your wait interval long enough?

You can set the wait interval as an option for the MQGET call. You should ensure that you are waiting long enough for a response.

- 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 retrieved, so you may need to reset these values in order to get another message successfully.

Also, check whether you can get other messages from the queue.

- Can other applications get messages from the queue?
- Has another application got exclusive access to the queue?

If you are unable to find anything wrong with the queue, and MQSeries 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.
- Did the application stop?
- Did the application complete correctly?
 - Look for evidence of an abnormal end on the system log and VSE/ESA console.
- Did the application commit its changes, or were they backed out?

If multiple transactions are serving the queue, they can conflict with one another. For example, suppose one transaction issues an MQGET call with a buffer length of zero to find out the length of the message, and then issues a specific MQGET call specifying the *MsgId* of that message. However, in the meantime, another transaction issues a successful MQGET call for that message, so the first application receives a reason 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.

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 that contain unexpected or corrupted information."

Messages that 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 onto the queue, changed?
 - Ensure that all changes are simultaneously reflected on all systems that need to be aware of the change.
 - For example, the format of the message data may have been changed, in which case both applications must be recompiled to pick up the changes. If one application has not been recompiled, the data will appear corrupt to the other.
- 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 your application has used an alias queue, check that the alias points to the correct queue.
- 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?

System log

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

Problems with incorrect output when using distributed queues

If your application uses distributed queues, you should also consider the following points:

- Has MQSeries been correctly installed on both the sending and receiving systems, and correctly configured for distributed queuing?
- Are the links available between the two systems?

Check that both systems are available, and connected to MQSeries. Check that the connection between the two systems, and the channels between the two queue managers, are active.

- Is triggering set on in the sending system?
- Is the message you are waiting for a reply message from a remote system?

Check that triggering is activated in the remote system.

- Is the queue already full?

This could mean that an application was unable to put the required message onto the queue. If this is so, check if the message has been put onto the dead-letter queue.

The dead-letter queue header contains a reason or feedback code explaining why the message could not be put onto the target queue. See the *MQSeries Application Programming Reference* manual for information about the dead-letter queue 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.

- Are the channel definitions of the sending and receiving channels compatible?

For example, a mismatch in sequence number wrap stops the distributed queuing component. See the *MQSeries Intercommunication* book for more information about distributed queuing.

System log

MQSeries uses the SYSTEM.LOG queue defined in the global system definition as its primary message log and additional informational messages are output to the VSE/ESA console. Typically, these detail starting, stopping, and initializing MQSeries for VSE/ESA

If the SYSTEM.LOG queue is unavailable, the messages are directed to the CICS CSMT log. These messages should always be reviewed carefully for any error messages.

Dead-letter queues

Messages that cannot be delivered for some reason are placed on the dead-letter queue. You can check whether the queue contains any messages by using the MQMT transaction. If the queue contains messages, you can use the browse facility to browse messages on the queue using the MQGET call.

You must decide how to dispose of any messages found on the dead-letter queue, depending on the reasons for the messages being put on the queue.

Problems may occur if you do not have a dead-letter queue on each queue manager you are using.

Using MQSeries trace

MQSeries uses the CICS auxiliary trace facility. This should only be used with the advice of IBM Service Personnel.

Problem determination with clients

An MQI client application receives MQRC_* reason codes in the same way as non-client MQI applications. However, there are now additional reason codes for error conditions associated with clients. For example:

- Remote machine not responding
- Communications line error
- Invalid machine address

The most common time for errors to occur is when an application issues an MQCONN and receives the response MQRC_Q_MQR_NOT_AVAILABLE. An error message, written to the client log file, explains the cause of the error. Messages may also be logged at the server depending on the nature of the failure.

Terminating clients

Even though a client has terminated it is still possible for the process at the server to be holding its queues open. Normally, this will only be for a short time until the communications layer detects that the partner has gone.

Error messages with clients

When an error occurs with a client system, error messages are put into the error files associated with the server, if possible. If an error cannot be placed there, the client code attempts to place the error message in an error log in the root directory of the client machine.

OS/2 and UNIX systems clients

Error messages for OS/2 and UNIX systems clients are placed in the error logs on their respective MQSeries server systems. Typically, these files appear in the QMGRS\@SYSTEM\ERRORS directory.

Client problem determination

DOS and Windows clients

The location of the log file AMQERR01.LOG is set by the MQDATA environment variable. The default location, if not overridden by MQDATA, is:

C:\

Working in the DOS environment involves the environment variable MQDATA.

This is the default library used by the client code to store trace and error information; it also holds the directory name in which the qm.ini file is stored. (needed for NetBIOS setup). If not specified, it defaults to the C drive.

The names of the default files held in this library are:

AMQERR01.LOG For error messages.

AMQERR01.FDC For First Failure Data Capture messages.

Appendix A. MQSeries for VSE/ESA at a glance

Program number

- 5686-A06 MQSeries for VSE/ESA Version 2 Release 1 (Europe, the Middle East and Africa only).

Hardware requirements

- MQSeries Servers:
 - Any IBM System/370® or System/390® machine
 - Minimum system memory – normal memory supplied with machine
 - Minimum DASD = VSE library requirements + size of queues
 - VSE library requirements
 - 3380 = 3 cylinders
 - 3390 = 2 cylinders
 - FBA (Fixed Block Architecture) = 4500 blocks

Software requirements

Minimum supported levels are shown. Later levels, if any, will be supported unless otherwise stated.

- VSE/ESA 2.3 (5690-VSE) or later 2.x
- CICS/VSE 2.3 (5686-026)
- VTAM for VSE/ESA 4.2 (5666-363) or later 4.x
- or
- TCP/IP for VSE/ESA V1.3 (5686-A04)
- with PTFs:
 - NFS (from CSI) –PQ14716 UQ19196
 - NFS Configuration Dialog –PQ14718 UQ18722
 - LE Socket Support –PQ16251 UQ18646
- LE/VSE 1.4 Runtime library (5696-067)
- MQSeries Clients:
 - MQSeries for VSE/ESA supports clients which can be connected using TCP/IP.

Connectivity

Network protocols supported are SNA LU6.2 and TCP/IP.

- For SNA connectivity – VTAM for VSE/ESA V4.2
- For TCP/IP connectivity – TCP/IP for VSE/ESA V1.3

Overview

Compilers supported for MQSeries for VSE/ESA applications

- Programs can be written using C, COBOL or PL/I
- C programs can use the C for VSE V1.1 compiler (5686-A01)
- COBOL programs can use the COBOL for VSE compiler V1.1 (5686-068)
- PL/I programs can use the PL/I for VSE compiler V1.1 (5696-069)

Delivery

MQSeries for VSE/ESA is available on

- 3480 cartridge
- 4mm DAT tape
- 9-track 6250 bpi tape

Appendix B. CICS control table definitions

This appendix contains various sample entries for the CICS control tables.

Sample file control table entries

```

*-----*
* Licensed Materials - Property of IBM *
* 5686-A06 *
* (C) Copyright IBM Corp. 1998 *
* *
* US Government Users Restricted Rights - Use, duplication or *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
*-----*
*
*-----*
*          Start of MQSeries VSAM cluster definitions *
* *
* For performance reasons entries may be modified to add LSRPOOL *
* explicit specifications. *
*-----*
*
* system setup file
MQFSSET DFHFCT TYPE=DATASET,DATASET=MQFSSET, *
          ACCMETH=VSAM, *
          SERVREQ=(READ,BROWSE), *
          LOG=NO, *
          RSL=PUBLIC, *
          STRNO=5, *
          RECFORM=(FIXED,BLOCKED)
* configuration file
MQFCNFG DFHFCT TYPE=DATASET,DATASET=MQFCNFG, *
          ACCMETH=VSAM, *
          SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE), *
          LOG=YES, *
          RSL=PUBLIC, *
          STRNO=20, *
          RECFORM=(FIXED,BLOCKED)
*--example of queues (input followed by output)
MQFI001 DFHFCT TYPE=DATASET,DATASET=MQFI001, *
          ACCMETH=VSAM, *
          SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE), *
          RSL=PUBLIC, *
          LOG=YES, *
          STRNO=16, *
          RECFORM=(VARIABLE,BLOCKED)
MQF0001 DFHFCT TYPE=DATASET,DATASET=MQF0001, *
          ACCMETH=VSAM, *
          SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE), *
          LOG=YES, *
          RSL=PUBLIC, *
          STRNO=16, *
          RECFORM=(VARIABLE,BLOCKED)
MQFI002 DFHFCT TYPE=DATASET,DATASET=MQFI002, *
          ACCMETH=VSAM, *
          SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE), *
          RSL=PUBLIC, *
          LOG=YES, *
          STRNO=16, *
          RECFORM=(VARIABLE,BLOCKED)
MQF0002 DFHFCT TYPE=DATASET,DATASET=MQF0002, *

```

Sample DCT

```

                                ACCMETH=VSAM,
                                SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),
                                LOG=YES,
                                RSL=PUBLIC,
                                STRNO=16,
                                RECFORM=(VARIABLE,BLOCKED)
MQFI003 DFHFCT TYPE=DATASET,DATASET=MQFI003,
                                ACCMETH=VSAM,
                                SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),
                                LOG=YES,
                                RSL=PUBLIC,
                                STRNO=16,
                                RECFORM=(VARIABLE,BLOCKED)
MQF0003 DFHFCT TYPE=DATASET,DATASET=MQF0003,
                                ACCMETH=VSAM,
                                SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),
                                LOG=YES,
                                RSL=PUBLIC,
                                STRNO=16,
                                RECFORM=(VARIABLE,BLOCKED)
*--SYSTEM DEFINITIONS
MQFLOG DFHFCT TYPE=DATASET,DATASET=MQFLOG,
                                ACCMETH=VSAM,
                                SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),
                                LOG=YES,
                                RSL=PUBLIC,
                                STRNO=16,
                                RECFORM=(VARIABLE,BLOCKED)
MQFERR DFHFCT TYPE=DATASET,DATASET=MQFERR,
                                ACCMETH=VSAM,
                                SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),
                                LOG=YES,
                                RSL=PUBLIC,
                                STRNO=16,
                                RECFORM=(VARIABLE,BLOCKED)
MQFMON DFHFCT TYPE=DATASET,DATASET=MQFMON,
                                ACCMETH=VSAM,
                                SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),
                                LOG=NO,
                                RSL=PUBLIC,
                                STRNO=16,
                                RECFORM=(VARIABLE,BLOCKED)
*-----*
*           End of MQSeries VSAM cluster definitions           *
*-----*
```

Sample destination control table entry

Note: Entry MQER is required in order for MQSeries System error messages to be logged to the SYSTEM.LOG queue.

```

*-----*
* Licensed Materials - Property of IBM
* 5686-A06
* (C) Copyright IBM Corp. 1998
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
*-----*
*
*-----*
* START OF MQSERIES DCT ENTRIES
```

```

*-----*
MQER      DFHDCT TYPE=INTRA,
          RSL=PUBLIC,
          DESTID=MQER,
          DESTFAC=FILE,
          TRANSID=MQER,
          TRIGLEV=1
*-----*
*   END OF MQSERIES DCT ENTRIES
*-----*

```

Sample JCL to process MQPUTIL

```

* ** JOB JNM=MQJUTILY,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJUTILY - Execute VSE/ESA MQ/Series Batch Utility Program.
* -----*
*   I M P O R T A N T   I M P O R T A N T   I M P O R T A N T   *
*
*   Please change :
*
*       "* ** JOB" to "* $$ JOB"
*       "* ** LST" to "* $$ LST"
*       "* ** EOJ" to "* $$ EOJ"
*
* -----*
*   This job executes MQPUTIL to access the CONFIGURATION file
*
*   This file is a sample and needs modification to suit the
*   users environment.
*
* -----*
*   Licensed Materials - Property of IBM
*   5686-A06
*   (C) Copyright IBM Corp. 1998
*
*   US Government Users Restricted Rights - Use, duplication or
*   disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
* -----*
// DLBL CONFIG,'MQSERIES.MQFCNFG',,VSAM,CAT=MQMCAT
// EXEC IDCAMS,SIZE=AUTO
/*
/*          VERIFY VSAM FILE
/*
/*          VERIFY FILE(CONFIG)
/*
// LIBDEF PHASE,SEARCH=(PRD2.MQSERIES,PRD2.SCEEBASE)
// ASSGN SYS004,SYSIPT
// ASSGN SYS005,SYSLST
// EXEC MQPUTIL,SIZE=AUTO
*RESET MSN          00000002
*RESET CHECKPOINT  00000002
*PRINT RESOLUTIONS
*PRINT CONFIG
*PRINT LOG
/*
/&
* ** EOJ

```

Sample JCL file definition for CICS deck

```

*-----*
* Licensed Materials - Property of IBM *
* 5686-A06 *
* (C) Copyright IBM Corp. 1998 *
* *
* US Government Users Restricted Rights - Use, duplication or *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
*-----*
* Sample JCL file definition for CICS deck *
* The DLBL statements in this JCL correspond to entries in CICSFC *
* therefore if there are any new file ids to be added in here, *
* it must also be added into the corresponding JCL *
* *
* Fields marked with ?valid? and ?cat-name? must be changed to *
* suit customer own site specifications. *
*-----*
// DLBL MQFSSET,'MQSERIES.MQFSSET',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFCNFG,'MQSERIES.MQFCNFG',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFI001,'MQSERIES.MQFI001',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFI002,'MQSERIES.MQFI002',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFI003,'MQSERIES.MQFI003',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQF0001,'MQSERIES.MQF0001',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQF0002,'MQSERIES.MQF0002',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQF0003,'MQSERIES.MQF0003',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFERR,'MQSERIES.MQFERR',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFLOG,'MQSERIES.MQFLOG',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
// DLBL MQFMON,'MQSERIES.MQFMON',0,VSAM,CAT=?cat-name?
// EXTENT ,?valid?
*-----*
* End of sample jcl file definition for cics deck *
*-----*

```

Sample JCL to create CICS CSD group

```

* ** JOB JNM=MQJCSD,CLASS=0,DISP=D
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJCSD Define resources for MQSeries for VSE/ESA to CICS CSD.
*-----*
* Please change :
*   "* ** JOB" to "* $$ JOB"
*   "* ** LST" to "* $$ LST"
*   "* ** EOJ" to "* $$ EOJ"
*-----*
* Create CICS CSD group for MQSeries VSE/ESA
* This file is a sample and may need modifications to suit the
* users environment (eg. Group name, or list name).
*-----*
* Licensed Materials - Property of IBM
* 5686-A06
* (C) Copyright IBM Corp. 1998
*

```



```

* US Government Users Restricted Rights - Use, duplication or      *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
* -----*
// EXEC DFHCSDUP
*-----*
*           Definitions for MQSeries VSE/ESA                       *
*-----*
*
DELETE GROUP(MQM)
*--
*--           Definitions of MQSeries Programs
*
DEFINE PROGRAM(MQBICIRH) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQBICITK) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC) *
RESIDENT(YES)
DEFINE PROGRAM(MQPMTM ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMCFG ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMON ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMOPR ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPDISP ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMSYS ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMQUE ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMCHN ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMSS ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMSC ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMMSN ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMISI ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMDEL ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMMOQ ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPMOC ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
*-- NON-ADMINISTRATOR
DEFINE PROGRAM(MQPAIP0 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPAIP1 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPAIP2 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSEND ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPRECV ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPRPRT ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPCKPT) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPQUE1 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPQUE2 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPECHO ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPINIT1) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPINIT2) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSSQ ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPCHK ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPERR ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPEXT1 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPFINQ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPQDEL ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSTOP) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSTART) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSREC ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPQREC ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSSMAP) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSSSET) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSENV ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPSCMD ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPVSAM ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(MQPTCPLN) GROUP(MQM) LANGUAGE(C) RSL(PUBLIC)
DEFINE PROGRAM(MQPTCPSV) GROUP(MQM) LANGUAGE(C) RSL(PUBLIC)
*-- MAPS
DEFINE MAPSET(MQMMTP ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMCFG ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMON ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMOPR ) GROUP(MQM) RSL(PUBLIC)

```

```

DEFINE MAPSET(MQMDISP ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMSYS ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMQUE ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMCHN ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMSS ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMSC ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMSN ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMSI ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMDL ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMMOQ ) GROUP(MQM) RSL(PUBLIC)
DEFINE MAPSET(MQMMOC ) GROUP(MQM) RSL(PUBLIC)
*-- TEST PROGRAMS
DEFINE PROGRAM(TTPTST1 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(TTPTST2 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE PROGRAM(TTPTST3 ) GROUP(MQM) LANGUAGE(COBOL) RSL(PUBLIC)
DEFINE MAPSET(TTMTST3 ) GROUP(MQM) RSL(PUBLIC)
*
*-- Definitions of MQSeries Transactions
*
DEFINE TRANSACTION(MQBI) GROUP(MQM) PROGRAM(MQBICITK)
DEFINE TRANSACTION(MQMT) GROUP(MQM) PROGRAM(MQPMTTP)
DEFINE TRANSACTION(MQMC) GROUP(MQM) PROGRAM(MQPMCFG)
DEFINE TRANSACTION(MQMO) GROUP(MQM) PROGRAM(MQPMOPR)
DEFINE TRANSACTION(MQMM) GROUP(MQM) PROGRAM(MQPMMON)
DEFINE TRANSACTION(MQBQ) GROUP(MQM) PROGRAM(MQPDISP)
DEFINE TRANSACTION(MQMS) GROUP(MQM) PROGRAM(MQPMSYS)
DEFINE TRANSACTION(MQDS) GROUP(MQM) PROGRAM(MQPMSYS)
DEFINE TRANSACTION(MQMQ) GROUP(MQM) PROGRAM(MQPMQUE)
DEFINE TRANSACTION(MQDQ) GROUP(MQM) PROGRAM(MQPMQUE)
DEFINE TRANSACTION(MQMH) GROUP(MQM) PROGRAM(MQPMCHN)
DEFINE TRANSACTION(MQDH) GROUP(MQM) PROGRAM(MQPMCHN)
DEFINE TRANSACTION(MQMA) GROUP(MQM) PROGRAM(MQPMSS)
DEFINE TRANSACTION(MQMB) GROUP(MQM) PROGRAM(MQPMSC)
DEFINE TRANSACTION(MQMR) GROUP(MQM) PROGRAM(MQPMMSN)
DEFINE TRANSACTION(MQMI) GROUP(MQM) PROGRAM(MQPMSI)
DEFINE TRANSACTION(MQMD) GROUP(MQM) PROGRAM(MQPMDEL)
DEFINE TRANSACTION(MQQM) GROUP(MQM) PROGRAM(MQPMMOQ)
DEFINE TRANSACTION(MQCM) GROUP(MQM) PROGRAM(MQPMOC)
DEFINE TRANSACTION(MQIT) GROUP(MQM) PROGRAM(MQPINIT1)
DEFINE TRANSACTION(MQ02) GROUP(MQM) PROGRAM(MQPAIP2)
DEFINE TRANSACTION(MQ01) GROUP(MQM) PROGRAM(MQPRECV)
DEFINE TRANSACTION(MQ03) GROUP(MQM) PROGRAM(MQPSEND)
DEFINE TRANSACTION(MQSS) GROUP(MQM) PROGRAM(MQPSSQ)
DEFINE TRANSACTION(MQSM) GROUP(MQM) PROGRAM(MQPSCCHK)
DEFINE TRANSACTION(MQER) GROUP(MQM) PROGRAM(MQPERR)
DEFINE TRANSACTION(MQQD) GROUP(MQM) PROGRAM(MQPQDEL)
DEFINE TRANSACTION(MQQA) GROUP(MQM) PROGRAM(MQPQDEL)
DEFINE TRANSACTION(MQST) GROUP(MQM) PROGRAM(MQPSTOP)
DEFINE TRANSACTION(MQSU) GROUP(MQM) PROGRAM(MQPSSSET)
DEFINE TRANSACTION(MQSE) GROUP(MQM) PROGRAM(MQPSENV)
DEFINE TRANSACTION(MQSR) GROUP(MQM) PROGRAM(MQPSREC)
DEFINE TRANSACTION(MQSQ) GROUP(MQM) PROGRAM(MQPQREC)
DEFINE TRANSACTION(MQTL) GROUP(MQM) PROGRAM(MQPTCLN)
DEFINE TRANSACTION(MQRG) GROUP(MQM) PROGRAM(MQPVSAM)
*-- Test Transactions
DEFINE TRANSACTION(TST1) GROUP(MQM) PROGRAM(TTPTST1)
DEFINE TRANSACTION(TST2) GROUP(MQM) PROGRAM(TTPTST2)
DEFINE TRANSACTION(TST3) GROUP(MQM) PROGRAM(TTPTST3)
DEFINE TRANSACTION(MQCL) GROUP(MQM) PROGRAM(MQPCMD)
*-- Add MQSeries group to the standard VSE/ESA list.
ADD GROUP(MQM) LIST(VSELIST)
/*
/&
* ** E0J

```

Appendix C. Application Programming Reference

This appendix should be used in conjunction with the *MQSeries Application Programming Reference* manual.

Structure data types

The following structure data types are supported by MQSeries for VSE/ESA V2.1:

MQDLH	Dead letter header
MQGMO	Get message options
MQMD	Message descriptor
MQOD	Object descriptor
MQPMO	Put message options
MQTM	Trigger message 2

The declarations of these structures are as described in the *MQSeries Application Programming Reference*, with the following exceptions.

MQDLH – Dead-letter header

Fields

Version (MQLONG)

Structure version number.

The value must be:

MQDLH_VERSION_1

Version number for dead-letter header structure.

Reason (MQLONG)

Reason message arrived on dead-letter (undelivered-message) queue.

This identifies the reason why the message was placed on the dead-letter queue instead of on the original destination queue. It should be one of the MQRC_* values (for example, MQRC_Q_FULL).

The initial value of this field is MQRC_NONE.

PutApplType (MQLONG)

Type of application that put message on dead-letter (undelivered-message) queue.

This field has the same meaning as the *PutApplType* field in the message descriptor MQMD.

The value used by MQSeries for VSE/ESA is MQAT_CICS_VSE.

PutApplName (MQCHAR28)

Name of application that put message on dead-letter (undelivered-message) queue.

The format of the name is an eight character applid, followed by a four character tranid.

MQMD – Message descriptor

MQGMO – Get message options

The fields in the Version 1 structure only are supported.

Fields

Version (MQLONG)

Structure version number.

The value must be:

MQGMO_VERSION_1

Version-1 get-message options structure.

Options (MQLONG)

Options that control the action of MQGET.

The following options are supported:

MQGMO_WAIT

MQGMO_NO_WAIT

MQGMO_SYNCPOINT

MQGMO_BROWSE_FIRST

MQGMO_BROWSE_NEXT

MQGMO_ACCEPT_TRUNCATED_MSG

MQGMO_MSG_UNDER_CURSOR

MQGMO_LOCK

MQGMO_UNLOCK

Signal1 (MQLONG)

Signal.

This is a reserved field; its value is not significant.

Signal2 (MQLONG)

Signal identifier.

This is a reserved field; its value is not significant.

MQMD – Message descriptor

The fields in the Version 1 structure only are supported.

Fields

Version (MQLONG)

Structure version number.

The value must be:

MQMD_VERSION_1

Version-1 message descriptor structure.

Report (MQLONG)

Options for report messages.

This is a reserved field. The value must be zero.

MsgType (MQLONG)

Message type.

This indicates the type of the message. The value must be one of the following:

MQMT_DATAGRAM

Message not requiring a reply.

MQMT_REQUEST

Message requiring a reply.

MQMT_REPLY

Reply to an earlier request message.

MQMT_REPORT

Report message.

There are no application-defined values.

Expiry (MQLONG)

Message lifetime.

This is a reserved field. The value must be –1.

Feedback (MQLONG)

Feedback or reason code.

Feedback codes are grouped as follows:

MQFB_NONE

No feedback provided.

MQFB_SYSTEM_FIRST

Lowest value for system-generated feedback.

MQFB_SYSTEM_LAST

Highest value for system-generated feedback.

MQFB_APPL_FIRST

Lowest value for application-generated feedback.

MQFB_APPL_LAST

Highest value for application-generated feedback.

Applications that generate report messages should not use feedback codes in the system range (other than MQFB_QUIT), unless they wish to simulate report messages generated by the queue manager or message channel agent.

A special feedback code is:

MQFB_QUIT

Application should end.

CodedCharSetId (MQLONG)

Coded character set identifier.

The following value only is defined:

MQCCSI_Q_MGR

Queue manager's coded character set identifier.

MQMD – Message descriptor

Format (MQCHAR8)

Format name.

The queue manager built-in formats are:

MQFMT_NONE

No format name.

MQFMT_ADMIN

Command server request/reply message.

MQFMT_CICS

CICS information header.

MQFMT_DEAD_LETTER_HEADER

Dead-letter header.

Priority (MQLONG)

Message priority.

There is no special value for this field.

Persistence (MQLONG)

Message persistence.

For the MQPUT and MQPUT1 calls, the value must be one of the following:

MQPER_PERSISTENT

Message is persistent.

MQPER_NOT_PERSISTENT

Message is not persistent.

CorrelId (MQBYTE24)

Correlation identifier.

The following special value may be used:

MQCI_NONE

No correlation identifier is specified.

BackoutCount (MQLONG)

Backout counter.

This is a reserved field.

UserIdentifier (MQCHAR12)

User identifier.

This is a reserved field.

AccountingToken (MQBYTE32)

Accounting token.

This is a reserved field.

ApplIdentityData (MQCHAR32)

Application data relating to identity.

This is a reserved field.

PutApplType (MQLONG)

Type of application that put the message.

This is a reserved field.

PutApplName (MQCHAR28)

Name of application that put the message.

This is a reserved field.

PutDate (MQCHAR8)

Date when message was put.

This is a reserved field.

PutTime (MQCHAR8)

Time when message was put.

This is a reserved field.

ApplOriginData (MQCHAR4)

Application data relating to origin.

This is a reserved field.

MQOD – Object descriptor

The MQOD structure is used to specify a queue object. The fields in the Version 1 structure only are supported.

Fields

Version (MQLONG)

Structure version number.

The value must be:

MQOD_VERSION_1

Version-1 object descriptor structure.

ObjectType (MQLONG)

Object type.

Type of object being named in *ObjectName*. The value must be:

MQOT_Q

Queue.

DynamicQName (MQCHAR48)

Dynamic queue name.

This is a reserved field.

AlternateUserId (MQCHAR12)

Alternate user identifier.

This is a reserved field.

MQTM – Trigger message

MQPMO – Put message options

The fields in the Version 1 structure only are supported.

Fields

Version (MQLONG)

Structure version number.

The value must be:

MQPMO_VERSION_1

Version-1 put-message options structure.

Options (MQLONG)

Options that control the action of MQPUT and MQPUT1.

The following option only is supported:

MQPMO_SYNCPOINT

Context (MQHOBJ)

Object handle of input queue.

This is a reserved field.

KnownDestCount (MQLONG)

Number of messages sent successfully to local queues.

This is a reserved field.

UnknownDestCount (MQLONG)

Number of messages sent successfully to remote queues.

This is a reserved field.

InvalidDestCount (MQLONG)

Number of messages that could not be sent.

This is a reserved field.

ResolvedQName (MQCHAR48)

Resolved name of destination queue.

This is a reserved field.

MQTM – Trigger message

On the MQSeries for VSE/ESA platform, triggers are invoked by the queue manager using either the transaction ID code, or program ID code, in the queue definition.

The transaction ID, or program ID, determines if the trigger is invoked using an EXEC CICS START, or EXEC CICS LINK, program respectively.

Trigger programs using the START mechanism can use the EXEC CICS RETRIEVE program to retrieve the trigger data structure. Programs invoked by the LINK mechanism can retrieve the MQTM structure in the DFHCOMMAREA.

Fields*Version* (MQLONG)

Structure version number.

The value must be:

MQTM_VERSION_1

Version number for trigger message structure.

ProcessName (MQCHAR48)

Name of process object.

This is a reserved field.

TriggerData (MQCHAR64)

Trigger data.

On MQSeries for VSE/ESA this is a 13-byte field, consisting of:

4-byte transaction ID code

8-byte program ID code

1-byte trigger-event flag

ApplType (MQLONG)

Application type.

On MQSeries for VSE/ESA *ApplType* has the following standard value:

MQAT_CICS_VSE

MQSeries for VSE/ESA application.

The initial value of this field is 0.

ApplId (MQCHAR256)

Application identifier.

On MQSeries for VSE/ESA *ApplId* is:

- A CICS transaction ID (for MQAT_CICS_VSE applications).

EnvData (MQCHAR128)

Environment data.

This is a reserved field.

UserData (MQCHAR128)

User data.

This is a reserved field.

MQI calls

This section identifies the MQI calls supported by MQSeries for VSE/ESA, which are:

MQCLOSE	Close object
MQCONN	Connect queue manager
MQDISC	Disconnect queue manager
MQGET	Get message
MQINQ	Inquire about object attributes
MQOPEN	Open object
MQPUT	Put message
MQPUT1	Put one message
MQSET	Set object attributes

These calls work as described in the *MQSeries Application Programming Reference* manual, except as described in the remainder of this section.

MQCLOSE – Close object

Parameters

Options (MQLONG) – input

Options that control the action of MQCLOSE.

The *Options* parameter controls how the object is closed.

The following option only is supported, and must be specified:

MQCO_NONE

No optional close processing required.

Reason (MQLONG) – output

Reason code qualifying *CompCode*.

If *CompCode* is MQCC_FAILED:

MQRC_CONNECTION_BROKEN

(2009, X'7D9') Connection to queue manager lost.

MQRC_HCONN_ERROR

(2018, X'7E2') Connection handle not valid.

MQRC_HOBJ_ERROR

(2019, X'7E3') Object handle not valid.

MQRC_OPTIONS_ERROR

(2046, X'7FE') Options not valid or not consistent.

MQRC_STORAGE_NOT_AVAILABLE

(2071, X'817') Insufficient storage available.

MQCONN – Connect queue manager

The name specified must be the name of a local queue manager. No substitution through an alias queue manager can be used.

Parameters

QMgrName (MQCHAR48) – input
Name of queue manager.

The name specified must be the name of a local queue manager. In MQSeries for VSE/ESA there is only one queue manager in a CICS partition.

Reason (MQLONG) – output
Reason code qualifying *CompCode*.

If *CompCode* is MQCC_FAILED:

MQRC_MAX_CONNS_LIMIT_REACHED
(2025, X'7E9') Maximum number of connections reached.
MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Queue manager name not valid or not known.
MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Queue manager not available for connection.
MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Insufficient storage available.
MQRC_UNEXPECTED_ERROR
(2195, X'893') Unexpected error occurred.

MQDISC – Disconnect queue manager**Parameters**

CompCode (MQLONG) – output
Completion code.

It is one of the following:

MQCC_OK
Successful completion.
MQCC_FAILED
Call failed.

Reason (MQLONG) – output
Reason code qualifying *CompCode*.

If *CompCode* is MQCC_FAILED:

MQRC_ADAPTER_DISC_LOAD_ERROR
(2138, X'85A') Unable to load adapter disconnection module.
MQRC_CONNECTION_BROKEN
(2009, X'7D9') Connection to queue manager lost.
MQRC_HCONN_ERROR
(2018, X'7E2') Connection handle not valid.
MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Insufficient storage available.
MQRC_UNEXPECTED_ERROR
(2195, X'893') Unexpected error occurred.

MQGET – Get message

Parameters

Hobj (MQHOBJ) – input
Object handle.

The queue must have been opened with one or more of the following options:

MQOO_INPUT_SHARED
MQOO_INPUT_EXCLUSIVE
MQOO_BROWSE

Reason (MQLONG) – output
Reason code qualifying *CompCode*.

The reason codes listed below are the ones that the queue manager can return for the *Reason* parameter.

If *CompCode* is MQCC_WARNING:

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Truncated message returned (processing completed).
MQRC_TRUNCATED_MSG_FAILED
(2080, X'820') Truncated message returned (processing not completed).

If *CompCode* is MQCC_FAILED:

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Buffer length parameter not valid.
MQRC_CONNECTION_BROKEN
(2009, X'7D9') Connection to queue manager lost.
MQRC_FILE_SYSTEM_ERROR
(2216, X'8A8') Queuer received file error.
MQRC_GET_INHIBITED
(2016, X'7E0') Gets inhibited for the queue.
MQRC_GMO_ERROR
(2186, X'88A') Get-message options structure not valid.
MQRC_HCONN_ERROR
(2018, X'7E2') Connection handle not valid.
MQRC_HOBJ_ERROR
(2019, X'7E3') Object handle not valid.
MQRC_MD_ERROR
(2026, X'7EA') Message descriptor not valid.
MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') No message available.
MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Browse cursor not positioned on message.
MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Queue not open for browse.
MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Queue not open for input.
MQRC_OPTIONS_ERROR
(2046, X'7FE') Options not valid or not consistent.

MQRC_STORAGE_NOT_AVAILABLE
 (2071, X'817') Insufficient storage available.
 MQRC_WAIT_INTERVAL_ERROR
 (2090, X'82A') Wait interval in MQGMO not valid.

MQINQ – Inquire about object attributes**Parameters**

Selectors (MQLONG×*SelectorCount*) – input
 Array of attribute selectors.

MQSeries for VSE/ESA supports only the following selectors:

Selectors for all types of queue

MQIA_DEF_PERSISTENCE
 Default message persistence.
 MQIA_INHIBIT_PUT
 Whether put operations are allowed.
 MQIA_Q_DESC
 Queue description.
 MQIA_Q_NAME
 Queue name.
 MQIA_Q_TYPE
 Queue type.

Selectors for local queues

MQCA_CREATION_DATE
 Queue creation date (MQ_CREATION_DATE_LENGTH).
 MQCA_CREATION_TIME
 Queue creation time (MQ_CREATION_TIME_LENGTH).
 MQCA_INITIATION_Q_NAME
 Initiation queue name (MQ_Q_NAME_LENGTH).
 MQIA_CURRENT_Q_DEPTH
 Number of messages on queue.
 MQIA_DEFINITION_TYPE
 Queue definition type.
 MQIA_INHIBIT_GET
 Whether get operations are allowed.
 MQIA_MAX_MSG_LENGTH
 Maximum message length.
 MQIA_MAX_Q_DEPTH
 Maximum number of messages allowed on queue.
 MQIA_OPEN_INPUT_COUNT
 Number of MQOPEN calls that have the queue open for input.
 MQIA_OPEN_OUTPUT_COUNT
 Number of MQOPEN calls that have the queue open for output.
 MQIA_TRIGGER_CONTROL
 Trigger control.
 MQIA_TRIGGER_TYPE
 Trigger type.
 MQIA_USAGE
 Usage.

Selectors for local definitions of remote queues

MQCA_REMOTE_Q_MGR_NAME

Name of remote queue manager
(MQ_Q_MGR_NAME_LENGTH).

MQCA_REMOTE_Q_NAME

Name of remote queue as known on remote queue manager
(MQ_Q_NAME_LENGTH).

MQCA_XMIT_Q_NAME

Name of local transmission queue.

Selectors for alias queues

MQCA_BASE_Q_NAME

Name of queue that alias resolves to
(MQ_Q_NAME_LENGTH).

MQIA_INHIBIT_GET

Whether get operations are allowed.

Reason (MQLONG) – output

Reason code qualifying *CompCode*.

If *CompCode* is MQCC_FAILED:

MQRC_CHAR_ATTR_LENGTH_ERROR

(2006, X'7D6') Length of character attributes not valid.

MQRC_CHAR_ATTRS_ERROR

(2007, X'7D7') Character attributes string not valid.

MQRC_CONNECTION_BROKEN

(2009, X'7D9') Connection to queue manager lost.

MQRC_HCONN_ERROR

(2018, X'7E2') Connection handle not valid.

MQRC_HOBJ_ERROR

(2019, X'7E3') Object handle not valid.

MQRC_INT_ATTR_COUNT_ERROR

(2021, X'7E5') Count of integer attributes not valid.

MQRC_INT_ATTRS_ARRAY_ERROR

(2023, X'7E7') Integer attributes array not valid.

MQRC_NOT_OPEN_FOR_INQUIRE

(2038, X'7F6') Queue not open for inquire.

MQRC_SELECTOR_COUNT_ERROR

(2065, X'811') Count of selectors not valid.

MQRC_SELECTOR_ERROR

(2067, X'813') Attribute selector not valid.

MQRC_SELECTOR_LIMIT_EXCEEDED

(2066, X'812') Count of selectors too big.

MQRC_STORAGE_NOT_AVAILABLE

(2071, X'817') Insufficient storage available.

MQRC_UNEXPECTED_ERROR

(2195, X'893') Unexpected error occurred.

MQOPEN – Open object

The MQOPEN call establishes access to an object. The following types of object are valid:

- Queue
- Queue manager

Parameters

Options (MQLONG) – input

Options that control the action of MQOPEN.

The following options apply and you must specify at least one of these. However, you cannot specify the two input options together.

```
MQOO_BROWSE
MQOO_INPUT_SHARED
MQOO_INPUT_EXCLUSIVE
MQOO_INQUIRE
MQOO_OUTPUT
```

The options are valid for alias, local, and remote queues, as described in the *MQSeries Application Programming Reference* manual.

CompCode (MQLONG) – output

Completion code.

It is one of the following:

```
MQCC_OK
    Successful completion.
MQCC_FAILED
    Call failed.
```

Reason (MQLONG) – output

Reason code qualifying *CompCode*.

If *CompCode* is MQCC_FAILED:

```
MQRC_ALIAS_BASE_Q_TYPE_ERROR
    (2001, X'7D1') Alias base queue not a valid type.
MQRC_CONNECTION_BROKEN
    (2009, X'7D9') Connection to queue manager lost.
MQRC_HANDLE_NOT_AVAILABLE
    (2017, X'7E1') No more handles available.
MQRC_HCONN_ERROR
    (2018, X'7E2') Connection handle not valid.
MQRC_OBJECT_IN_USE
    (2042, X'7FA') Object already open with conflicting options.
MQRC_OBJECT_TYPE_ERROR
    (2043, X'7FB') Object type not valid.
MQRC_OD_ERROR
    (2044, X'7FC') Object descriptor structure not valid.
MQRC_OPTION_NOT_VALID_FOR_TYPE
    (2045, X'7FD') Option not valid for object type.
MQRC_OPTIONS_ERROR
    (2046, X'7FE') Options not valid or not consistent.
MQRC_STORAGE_NOT_AVAILABLE
    (2071, X'817') Insufficient storage available.
```

MQPUT – Put message

MQRC_UNEXPECTED_ERROR
(2195, X'893') Unexpected error occurred.
MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822') Unknown alias base queue.
MQRC_UNKNOWN_OBJECT_NAME
(2085, X'825') Unknown object name.
MQRC_UNKNOWN_OBJECT_Q_MGR
(2086, X'826') Unknown object queue manager.
MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827') Unknown remote queue manager.

MQPUT – Put message

Parameters

Reason (MQLONG) – output

Reason code qualifying *CompCode*.

If *CompCode* is MQCC_WARNING:

MQRC_PRIORITY_EXCEEDS_MAXIMUM
(2049, X'801') Message Priority exceeds maximum value supported.

If *CompCode* is MQCC_FAILED:

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Buffer length parameter not valid.
MQRC_CONNECTION_BROKEN
(2009, X'7D9') Connection to queue manager lost.
MQRC_EXPIRY_ERROR
(2013, X'7DD') Expiry time not valid.
MQRC_FEEDBACK_ERROR
(2014, X'7DE') Feedback code not valid.
MQRC_HCONN_ERROR
(2018, X'7E2') Connection handle not valid.
MQRC_HOBJ_ERROR
(2019, X'7E3') Object handle not valid.
MQRC_MD_ERROR
(2026, X'7EA') Message descriptor not valid.
MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') Missing reply-to queue.
MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Message length greater than maximum for queue.
MQRC_MSG_TYPE_ERROR
(2029, X'7ED') Message type in message descriptor not valid.
MQRC_NOT_OPEN_FOR_OUTPUT
(2039, X'7F7') Queue not open for output.
MQRC_OPTIONS_ERROR
(2046, X'7FE') Options not valid or not consistent.
MQRC_PERSISTENCE_ERROR
(2047, X'7FF') Persistence not valid.
MQRC_PMO_ERROR
(2173, X'87D') Put-message options structure not valid.
MQRC_PRIORITY_ERROR
(2050, X'802') Message priority not valid.

MQRC_PUT_INHIBITED
 (2051, X'803') Put calls inhibited for the queue.

MQRC_Q_FULL
 (2053, X'805') Queue already contains maximum number of messages.

MQRC_Q_SPACE_NOT_AVAILABLE
 (2056, X'808') No space available on disk for queue.

MQRC_REPORT_OPTIONS_ERROR
 (2061, X'80D') Report options in message descriptor not valid.

MQRC_STORAGE_NOT_AVAILABLE
 (2071, X'817') Insufficient storage available.

MQRC_UNEXPECTED_ERROR
 (2195, X'893') Unexpected error occurred.

MQRC_UNKNOWN_CCSID
 (2115, X'843') Unknown CCSID.

MQPUT1 – Put one message

Parameters

Reason (MQLONG) – output

Reason code qualifying *CompCode*.

If *CompCode* is MQCC_WARNING:

MQRC_PRIORITY_EXCEEDS_MAXIMUM

(2049, X'801') Message Priority exceeds maximum value supported.

If *CompCode* is MQCC_FAILED:

MQRC_ALIAS_BASE_Q_TYPE_ERROR

(2001, X'7D1') Alias base queue not a valid type.

MQRC_BUFFER_LENGTH_ERROR

(2005, X'7D5') Buffer length parameter not valid.

MQRC_CONNECTION_BROKEN

(2009, X'7D9') Connection to queue manager lost.

MQRC_EXPIRY_ERROR

(2013, X'7DD') Expiry time not valid.

MQRC_FEEDBACK_ERROR

(2014, X'7DE') Feedback code not valid.

MQRC_HANDLE_NOT_AVAILABLE

(2017, X'7E1') No more handles available.

MQRC_HCONN_ERROR

(2018, X'7E2') Connection handle not valid.

MQRC_MD_ERROR

(2026, X'7EA') Message descriptor not valid.

MQRC_MISSING_REPLY_TO_Q

(2027, X'7EB') Missing reply-to queue.

MQRC_MSG_TOO_BIG_FOR_Q

(2030, X'7EE') Message length greater than maximum for queue.

MQRC_MSG_TYPE_ERROR

(2029, X'7ED') Message type in message descriptor not valid.

MQRC_OBJECT_TYPE_ERROR

(2043, X'7FB') Object type not valid.

MQRC_OD_ERROR

(2044, X'7FC') Object descriptor structure not valid.

MQSET – Set object attributes

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options not valid or not consistent.

MQRC_PERSISTENCE_ERROR
(2047, X'7FF') Persistence not valid.

MQRC_PMO_ERROR
(2173, X'87D') Put-message options structure not valid.

MQRC_PRIORITY_ERROR
(2050, X'802') Message priority not valid.

MQRC_PUT_INHIBITED
(2051, X'803') Put calls inhibited for the queue.

MQRC_Q_FULL
(2053, X'805') Queue already contains maximum number of messages.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808') No space available on disk for queue.

MQRC_REPORT_OPTIONS_ERROR
(2061, X'80D') Report options in message descriptor not valid.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Insufficient storage available.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Unexpected error occurred.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822') Unknown alias base queue.

MQRC_UNKNOWN_OBJECT_NAME
(2085, X'825') Unknown object name.

MQRC_UNKNOWN_OBJECT_Q_MGR
(2086, X'826') Unknown object queue manager.

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827') Unknown remote queue manager.

MQRC_UNKNOWN_CCSID
(2115, X'843') Unknown CCSID.

MQSET – Set object attributes

Parameters

Hconn (MQHCONN) – input
Connection handle.

The value of *Hconn* was returned by a previous MQCONN call.

Selectors (MQLONG×*SelectorCount*) – input
Array of attribute selectors.

MQSeries for VSE/ESA supports only the following selectors.

Selectors for all types of queue

MQIA_INHIBIT_PUT
Whether put operations are allowed.

Selectors for local queues

MQIA_INHIBIT_GET
Whether get operations are allowed.

Selectors for alias queues

MQIA_INHIBIT_GET

Whether get operations are allowed.

Reason (MQLONG) – output

Reason code qualifying *CompCode*.

If *CompCode* is MQCC_FAILED:

MQRC_CHAR_ATTR_LENGTH_ERROR

(2006, X'7D6') Length of character attributes not valid.

MQRC_CHAR_ATTRS_ERROR

(2007, X'7D7') Character attributes string not valid.

MQRC_CICS_WAIT_FAILED

(2140, X'85C') Wait request rejected by CICS.

MQRC_CONNECTION_BROKEN

(2009, X'7D9') Connection to queue manager lost.

MQRC_HCONN_ERROR

(2018, X'7E2') Connection handle not valid.

MQRC_HOBJ_ERROR

(2019, X'7E3') Object handle not valid.

MQRC_INHIBIT_VALUE_ERROR

(2020, X'7E4') Value for inhibit-get or inhibit-put queue attribute not valid.

MQRC_INT_ATTR_COUNT_ERROR

(2021, X'7E5') Count of integer attributes not valid.

MQRC_INT_ATTRS_ARRAY_ERROR

(2023, X'7E7') Integer attributes array not valid.

MQRC_NOT_OPEN_FOR_SET

(2040, X'7F8') Queue not open for set.

MQRC_OBJECT_CHANGED

(2041, X'7F9') Object definition changed since opened.

MQRC_Q_MGR_NAME_ERROR

(2058, X'80A') Queue manager name not valid or not known.

MQRC_Q_MGR_NOT_AVAILABLE

(2059, X'80B') Queue manager not available for connection.

MQRC_SELECTOR_COUNT_ERROR

(2065, X'811') Count of selectors not valid.

MQRC_SELECTOR_ERROR

(2067, X'813') Attribute selector not valid.

MQRC_SELECTOR_LIMIT_EXCEEDED

(2066, X'812') Count of selectors too big.

MQRC_STORAGE_NOT_AVAILABLE

(2071, X'817') Insufficient storage available.

MQRC_UNEXPECTED_ERROR

(2195, X'893') Unexpected error occurred.

Attributes of MQSeries objects

In MQSeries for VSE/ESA, the attributes of all objects are as described in the *MQSeries Application Programming Reference* manual, with the following exception:

- Attributes of process definitions do not apply

The platform constant MQAT_CICS_VSE applies, value 10L.

Reason codes

In MQSeries for VSE/ESA, reason codes (MQRC_) are as described in the *MQSeries Application Programming Reference* manual.

Appendix D. Application Programming Guidance

This appendix describes:

- Application program support
- Samples
- Syncpointing
- Triggers

Supporting application programs that use the MQI

MQSeries application programs need specific objects before they can run successfully. For example, Figure 43 shows an application that removes messages from a queue, processes them, and then sends some results to another queue on the same queue manager.

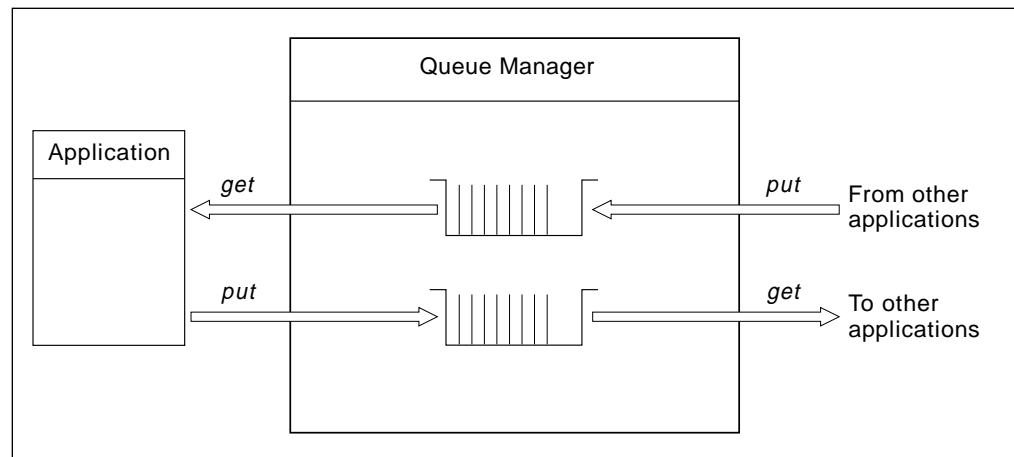


Figure 43. Queues, messages, and applications

Whereas applications can put (using MQPUT) messages on local or remote queues, they can only get (using MQGET) messages directly from local queues.

Before this application can run, these conditions must be satisfied:

- The queue manager must exist and be running.
- The first application queue, from which the messages are to be removed, must be defined.
- The second queue, on which the application puts the messages, must also be defined.
- The application must be able to connect to the queue manager. To do this the queue manager must be linked to the product code.
- The applications that put the messages on the first queue must also connect to a queue manager. If they are remote, they must also be set up with transmission queues and channels. This part of the system is not shown in Figure 43.

Sample source code provided

One COBOL-language sample trigger program, MQPECHO is provided with MQSeries for VSE/ESA. The source code for this program is shown in “Sample program MQPECHO.Z” on page 173, or it can be listed directly from the distribution files. Within the source code for MQPECHO, you can find examples that illustrate the use of the MQI calls in a trigger program.

In addition there are three sample programs, TTPST1, TTPST2, and TTPST3. COBOL language copybook files are provided with the distribution file in the PRD2.MQSERIES library. These files provide examples of all of the MQI calls.

Compiling your application program

The MQI calls are provided in the library PRD2.MQSERIES.

Compilation

Ensure that you include the PRD2.MQSERIES library as part of the application phase step.

Developing applications in the C and PL/I programming languages

For CICS, COBOL is the language in which the MQSeries interface is written. Applications written in COBOL for VSE have been thoroughly tested with MQSeries. Sample programs and copybooks are provided in COBOL for VSE.

However, for a variety of reasons, you may need to write in another programming language. In these cases, you must meet the requirements of the COBOL language interface.

There are no sample programs provided in any other language, however, there are equivalents to the COBOL copybooks to enable applications to be built in other languages.

For the PL/I programming language, there are two include files:

CMQEPP.P Declares the MQI calls and structures
CMQP.P Declares the MQI constants

For the C programming language, there is the one include file CMQC.H, which declares everything required.

Application design guidelines

One of the key benefits provided by MQSeries is the ability for a distributed application to be developed that is totally independent of the underlying network. This network independence means that there is no need for an application to be aware of:

- The lower levels of the communication protocols, or
- The physical location of other applications on the network.

In order to take full advantage of this network independence, you must choose the queue names used by the application with care.

In particular, you are recommended to use a single logical name only, in your application programs, to refer to each MQSeries queue. For the MQSeries calls, this means only the Queue_Name field is used to identify queues. The use of the queue's fully qualified name (which includes both the Queue_Name field and the Queue_Manager_Name field) is not recommended.

The same is true when addressing MQSeries queues. As the Queue_Manager_Name is typically associated with a particular system, its use implies knowledge of the physical network.

Note: You are strongly recommended to use the Queue_Name field as the only logical queue name. This usage maximizes application flexibility and network independence. The mapping of the queue name in this form to the proper network destination then becomes a configuration issue to be handled by the MQSeries system administrator.

Syncpoints and triggers

This section describes syncpoints and triggers.

- Syncpoints allow an application to perform a series of changes, where the changes are treated as though they were a single change. They are described in “Syncpoint considerations.”
- Triggers allow applications to be started automatically when messages arrive. They are described in “Triggers” on page 141.

Syncpoint considerations

Most applications need to access resources of one form or another, and a common requirement is to be able to make a coordinated set of changes to two or more resources.

“Coordinated” means that either all of the changes made to the resources take effect, or none of the changes takes effect. For some applications, queues need to be coordinated. Applications need to be able to get and put messages (and possibly update other resources, for example, databases), and know that either all of the operations take effect, or that none of the operations takes effect.

This set of coordinated operations is called a unit of work. An example of a unit of work is a debit and credit for a funds transfer in a financial application. Both operations must complete, or neither operation must complete, for a valid financial transaction to be completed.

Units of work: A unit of work starts when the first recoverable resource is affected. For message queuing, a unit of works starts when a message get or put occurs under syncpoint control.

The unit of work ends when either the application ends, or when the application declares a syncpoint.

If the unit of work is ended by an application ending, another unit of work can start. One instance of an application can be involved with several sequential units of work.

Design guidelines

When a syncpoint is declared, any party (applications and the queue manager) that has interest in the unit of work can vote “yes” to commit the work, or “no”, to back out of the unit of work.

Applications declare syncpoints, and register their votes, by issuing an environment-dependent call. It is advisable that an application should process CICS SYNCPOINT prior to invoking an MQCLOSE call.

Participation of the MQGET, MQPUT, and MQPUT1 calls in the current unit of work is determined by the environment.

Distributed units of work (involving more than one queue manager) are not supported. A unit of work can contain queuing operations at only one instance of the queue manager.

If a message is put to a remote queue (that is, one on another queuing system), the action of the put request can be within the unit of work on the system that puts the message but the arrival of the message on the target (remote) queue is outside its scope.

The get request for the message on the remote queue can be within the scope of work on that system, but the two units of work are not related by the queue manager.

Putting messages within a unit of work: If an MQPUT or MQPUT1 call participates in the current unit of work, the message is not available for retrieval from the target queue, between the completion of the MQPUT call and the successful completion of the unit of work. The only exception to this rule is if the target queue is within the same unit of work as the one within which it was put.

Only when, and if, the unit of work is committed successfully does the message become generally available.

Any errors detected by the queue manager when the message is put are returned to the application immediately, by means of the completion code and reason code parameters. Errors that can be detected in this way include:

- Message too large for queue
- Queue full
- Put requests inhibited for queue

Failure to put the message does not affect the status of the unit of work, because that message is not part of the unit of work. The application can still commit or backout of the unit of work as required.

However, should an application fail after a message was put successfully within a unit of work, the transaction is backed out.

Getting messages within a unit of work: If an MQGET call participates in the current unit of work, between the completion of the MQGET call and the successful completion of the unit of work, the message remains on the queue but becomes invisible.

Neither the application that retrieved the message, nor any other application serving the queue, can see or obtain the message again. If the unit of work is committed

successfully, the message is deleted from the queue. However, if the unit of work is backed out, the message is reinstated in the queue in its original position, and becomes available to the same or another application to retrieve.

Syncpoint and persistence: Only persistent messaging is supported. Persistent messages do not get deleted if the queue manager is restarted. Therefore, they are fully recovered when the queue manager is restarted. Syncpointing by the application causes these records to be in a logical unit of work. Any records that were syncpointed are still recovered if the queue manager is shutdown and restarted.

Syncpoint Rollback

If your application wants to undo what has been done since the beginning of the current logical unit of work, it has to issue the following command:

```
EXEC CICS SYNCPOINT ROLLBACK
```

This can have the following, non-desired, results:

- Monitoring shows incorrect queue depth values, because the queue manager is not aware of rollbacks. This value is correctly reset when stopping and restarting the queue manager.
- The queue depth and the last sequence number are not the same anymore. Even if a message has been rolled back, its message sequence number (MSN) is never used again. This is because other applications may have also put messages into the same queue. For example:

```
Transaction A writes Message number 5
Transaction B ..... 6
Transaction A ..... 7
Transaction C ..... 8
```

At this point the queue depth is 8. Assume Transaction A rolls back, in which case messages 5 and 7 will be never retrieved. Note that this is not an error. The queue depth is now 6, and the next MSN will be 9.

From an application point of view this has no impact at all, but can be surprising when using the MQMT dialogs.

Note: To be able to use SYNCPOINT ROLLBACK, you MUST use a CICS System LOG file, that is, define a CICS JCT.

Triggers

Some applications run continuously, and are always available to read a message when it arrives on the application's input queue. However, keeping the application active consumes system resources, even when the application is waiting for a message to arrive. This additional load on the system is not desirable. Instead of the application running continuously, the application is designed to run only when there are messages to be processed. The queue manager's triggering facility is used to help make this happen.

Overview of triggering

A local queue definition can have a trigger event associated with it when it is defined. This event is defined to activate the MQSeries trigger API Handler, that is, the MQ02 CICS Transaction.

The trigger API handler does either a CICS LINK to the application program or a CICS START to the application transaction. This is based on whether you defined a program name or a transaction name in the queue definition.

When an application program is entered, an information area is available. This area can be mapped by using the structure defined in the member CMQTMV.C:

1. If the trigger facility specified a program name, this area is passed using the COMMAREA.

To return to the API handler, you should issue an EXEC CICS RETURN.

2. If the trigger facility specified a transaction name, this information area can be accessed by issuing an EXEC CICS RETRIEVE command.

Before exiting from the program, you must issue an MQCLOSE command.

Note: In order to perform this function, this transaction ID must be unique in respect to any MQSeries system local queue. Essentially, the MQSeries system queue manager recognizes this transaction ID as a local queue being opened. When this queue is closed fully, this trigger event will be closed, allowing another trigger for this queue to be activated.

Trigger conditions

The queue manager activates a trigger event based on the event type defined for the current queue, against which the MQPUT operation has been requested.

Note: If a non-empty queue is stopped and restarted, the trigger condition suffices, regardless of the trigger event type.

The trigger API handler waits until this MQPUT request has been completed. This implies that the MQPUT request can be successful or unsuccessful, that is, rolled back. The activated trigger application program should perform an MQGET call.

If the result of this MQGET call is an empty condition, that is, MQRC_NO_MSG_AVAILABLE, the original application current logical unit-of-work has been rolled back. It is up to the application trigger program to determine whether to continue to wait or just end.

A trigger event type of "FIRST" generates a trigger event after the queue goes from an empty status to a non-empty one. Therefore, any application triggered in this manner must process the queue until the queue is empty.

A trigger event type of "EVERY" generates a trigger event after every MQPUT call has been completed, up to the maximum number of trigger events specified on the Extended Local Queue Configuration screen. See "Local queue extended definition screen" on page 51 for further information.

Defining a sender channel component

A sender channel component causes the channel to start if there are messages on the transmission queue to be sent to the remote node.

In contrast, a server channel component will not start unless started by a remote requester component, or by manual intervention, even when there are messages to be sent.

On the transmission queue for the sender channel, code the fields as follows:

- Usage Mode – T
- Trigger Enable – Y
- Trigger Type – E
- Max Trigger Starts – 1
- Transaction ID – <blanks>
- Program ID – MQPSEND
- Remote CID – <the name of the channel>

Note: MQSeries for VSE/ESA does not support requester channels.

Defining a program to be triggered

This technique is used when an application program is to receive messages from the MQSeries system queue manager in the manner described in “Overview of triggering” on page 142 for a CICS LINK.

- Usage Mode – N
- Trigger Enable – Y
- Trigger Type – E or F
- Max Trigger Starts – 1
- Transaction ID – <blanks>
- Program ID – <application program name>
- Remote CID – <blanks>

Defining a transaction to be triggered

“Overview of triggering” on page 142, for CICS START, provides details of how to trigger a program based on its transaction ID. Note, that the transaction should not be invoked outside the trigger mechanism. However, by defining a different transaction name with the same program name, the program can be invoked outside the trigger environment.

Code as follows in the queue definition:

- Usage Mode – N
- Trigger Enable – Y
- Trigger Type – E or F
- Max Trigger Starts – 1
- Transaction ID – <user Transaction>
- Program ID – <blanks>
- Remote CID – <blanks>

Appendix E. Sample programs

This appendix lists the COBOL-language sample programs that are supplied with MQSeries for VSE/ESA V2.1.

Sample program TTPTST2.Z

This program is a test facility for sending and receiving messages. It must be invoked by terminal input format as:

```
TST2 func nn queue-name
```

where:

TST1

Is the transaction ID.

func

Is any of the following functions:

```

BOTH   Put and get messages.
GET    Get messages.
GETD   Get and delete messages.
INQ    Invoke MQINQ to inquire about the attributes of queues.
PUT    Put messages.
PUTR   Put messages and send reply.
PUT1   Put and delete messages.

```

nn

Is the number of messages to be processed, from 01 through 99.

queue name

Is the name of the local or transmission queue to be processed.

For example, TST2 PUT 99 QUE1 puts 99 messages into a transmission queue named QUE1. All the messages read THIS IS A MESSAGE TEXT. Typing TST2 without parameters causes help instructions to be displayed.

For each function, there is a corresponding set of MQCONN, MQOPEN, MQCLOSE and MQDISC operations. In the preceding example, there are 99 connections and disconnections to the Queue Manager and 99 open and close operations to QUE1.

```

IDENTIFICATION DIVISION.
PROGRAM-ID.    TTPTST2.
AUTHOR.       IBM.

DATE-COMPILED.

*
* SUMMARY CHANGES:
*-----*
* CHANGE DATE   PGM   COMMENT
*-----*
*
*****
*             T E S T   2   - ONLY ONE OPEN AND CLOSE
*
*             A P P L I C A T I O N   I N T E R F A C E
*
*             I B M   M Q I
*-----*

* TTPTST2 - MQI APPLICATION TEST PROGRAM
*
* FUNCTIONS:  1. PERFORM NORMAL QUEUE PUT
*             2. TRY TO GET QUEUE INFO BACK
*
* COPYBOOKS: MQIVALUE - MQI RETURN CODES.
*             MQIERRWS - ERROR WS
*             MQIERRCD - ERROR CODE
*
* CALLS      : MQCONN  - CONNECT
*             MQOPEN  - OPEN
*             MQPUT   - PUT
*             MQGET   - GET
*             MQCLOSE - CLOSE
*             MQDISC  - DISCONNECT
*
* CALLED BY:  -- NONE --
*-----*
/
ENVIRONMENT DIVISION.

```

TTPTST2.Z

CONFIGURATION SECTION.
DATA DIVISION.

WORKING-STORAGE SECTION.
COPY COPYRWS.

```

*-----*
01 WS-WORK-FIELDS.
05 WS-IDX PIC S9(4) COMP VALUE ZERO.
05 WS-COUNT PIC S9(4) COMP VALUE ZERO.
05 WS-PROCESS-TIMES PIC 9(4) VALUE ZERO.
05 WS-DURATION-SECS PIC X(8) VALUE SPACES.
05 WS-PASS-MSG-LENGTH PIC S9(4) COMP VALUE ZERO.
05 WS-APPL-MSG-LENGTH PIC S9(8) COMP VALUE ZERO.
05 WS-ABSTIME PIC S9(15) COMP-3 VALUE ZERO.
05 WS-ABSTIME2 PIC S9(15) COMP-3 VALUE ZERO.
05 WS-DATE.
10 WS-DATE-CC PIC 99 VALUE ZERO.
10 WS-DATE-YYMMDD.
12 WS-DATE-YY PIC 99 VALUE ZERO.
12 WS-DATE-MM PIC 99 VALUE ZERO.
12 WS-DATE-DD PIC 99 VALUE ZERO.
05 WS-TIME-9 PIC 9(7) VALUE ZERO.
05 WS-TIME REDEFINES WS-TIME-9.
10 FILLER PIC 9.
10 WS-TIME-HHMMSS.
12 WS-TIME-HH PIC 99.
12 WS-TIME-MM PIC 99.
12 WS-TIME-SS PIC 99.
05 WS-FORMATTED-TIME.
10 WS-FORMAT-TIME-HH PIC X(02) VALUE SPACES.
10 FILLER PIC X(01) VALUE '.'.
10 WS-FORMAT-TIME-MM PIC X(02) VALUE SPACES.
10 FILLER PIC X(01) VALUE '.'.
10 WS-FORMAT-TIME-SS PIC X(02) VALUE SPACES.
05 WS-FORMATTED-DATE.
10 WS-FORMAT-DATE-MM PIC X(02) VALUE SPACES.
10 FILLER PIC X(01) VALUE '/'.
10 WS-FORMAT-DATE-DD PIC X(02) VALUE SPACES.
10 FILLER PIC X(01) VALUE '/'.
10 WS-FORMAT-DATE-YY PIC X(02) VALUE SPACES.

05 WS-QM-Q-NAME.
10 WS-QM-NAME PIC X(48) VALUE 'QM1 '.
10 WS-Q-NAME PIC X(48) VALUE 'QUEUE'.

05 WS-REPLY-Q PIC X(48) VALUE 'QUE1'.

05 WS-ERR-MSG-FLAG PIC X VALUE SPACES.
88 WS-ERR-MSG VALUE 'Y'.

05 WS-STARTED-FLAG PIC X VALUE SPACES.
88 WS-STARTED VALUE 'Y'.

05 WS-TIMESTAMP PIC X VALUE SPACES.
88 WS-PUT-TIMESTAMP VALUE 'Y'.

05 WS-TIMESTAMP-VALUE.
10 WS-TIMESTAMP-DATE PIC X(6) VALUE SPACES.
10 WS-TIMESTAMP-TIME PIC X(6) VALUE SPACES.
10 WS-TIMESTAMP-COUNT PIC 999 VALUE ZERO.

05 WS-STARTCODE PIC XX VALUE SPACE.
88 START-WITH-DATA VALUE 'SD'.
88 START-WITH-NO-DATA VALUE 'S '.

05 WS-END-OF-MESSAGES-FLAG PIC X VALUE SPACES.
88 WS-END-OF-MESSAGES VALUE 'Y'.

05 WS-TRUNCATED-MESSAGES-F PIC X VALUE SPACES.
88 WS-TRUNCATED-MESSAGES VALUE 'Y'.

*-----*
EJECT
*-----*
* ERROR VALUES
*-----*
01 WS-ERR.
* COPY MQIERR.
*-----*
* - BEGIN - *** COPYBOOK: MQIERR *** - BEGIN - *
*-----*

```

```

* ERROR MODULE CALLING PARAMETERS
*-----*
02 ERR-HANDLER-COMMAREA.
05 ERR-CURRENT-INFO.
10 ERR-COM-HANDLER PIC X(48) VALUE SPACES.
10 ERR-QUEUE PIC X(48) VALUE SPACES.
10 ERR-FILE PIC X(8) VALUE SPACES.
10 ERR-DETAIL PIC X(80) VALUE SPACES.
10 ERR-DETAIL2 PIC X(80) VALUE SPACES.
10 ERR-Q-CODE PIC S9(8) COMP VALUE ZERO.
10 FILLER PIC X(8) VALUE SPACES.

05 ERR-RESULTS.
10 ERR-CODE PIC 9(6) VALUE ZERO.
10 FILLER PIC XX VALUE SPACES.
10 ERR-PROGRAM PIC X(8) VALUE SPACES.
10 ERR-TRANID PIC X(4) VALUE SPACES.
10 ERR-TERMID PIC X(4) VALUE SPACES.
10 ERR-TASKNO PIC S9(7) COMP-3 VALUE ZERO.
10 ERR-ABSTIME PIC S9(15) COMP-3 VALUE ZERO.

10 ERR-DEBUG-EIBFN PIC XX VALUE SPACES.
10 ERR-DEBUG-EIBRCODE PIC X(6) VALUE LOW-VALUES.
10 ERR-DEBUG-EIBRSRCE PIC X(8) VALUE LOW-VALUES.
10 ERR-DEBUG-EIBRESP PIC S9(8) COMP VALUE ZEROS.
10 ERR-DEBUG-EIBRESP2 PIC S9(8) COMP VALUE ZEROS.
10 ERR-DEBUG-EIBERRCD PIC X(4) VALUE LOW-VALUES.
10 ERR-DEBUG-ABEND PIC X(4) VALUE SPACES.
10 FILLER PIC X(12) VALUE SPACES.

*-----*
* - END - *** COPYBOOK: MQIERR *** - END - *
*-----*
* COPY MQIERRC.
*-----*
* IBM MQSERIES COMMON ERROR CODES
*-----*
01 MSG-ERROR-MESSAGES.
05 ERR-NO-ENVIRONMENT PIC 9(6) VALUE 900000.

05 ERR-CICS-ERROR PIC 9(6) VALUE 800000.
05 ERR-CICS-INVALID-REQ PIC 9(6) VALUE 800010.
05 ERR-CICS-ILLOGIC PIC 9(6) VALUE 800011.
05 ERR-CICS-ERROR-CHECKPOINT PIC 9(6) VALUE 800090.
05 ERR-CICS-ABEND PIC 9(6) VALUE 800099.
05 ERR-CICS-FILE-NOTOPEN PIC 9(6) VALUE 801012.
05 ERR-CICS-DISABLE PIC 9(6) VALUE 801019.
05 ERR-CICS-NO-STORAGE PIC 9(6) VALUE 802000.
05 ERR-CICS-LENGTH-ERR PIC 9(6) VALUE 803001.
05 ERR-CICS-MAPFAIL PIC 9(6) VALUE 808000.
05 ERR-CICS-PGMIDERR PIC 9(6) VALUE 809000.
05 ERR-CICS-FILEID PIC 9(6) VALUE 809010.
05 ERR-CICS-NOFILE PIC 9(6) VALUE 809011.
05 ERR-CICS-IO-ERROR PIC 9(6) VALUE 809012.
05 ERR-CICS-TRANIDERR PIC 9(6) VALUE 809050.

05 ERR-COM-FREE-ERROR PIC 9(6) VALUE 501001.
05 ERR-COM-EIB-ERROR PIC 9(6) VALUE 501002.
05 ERR-COM-STAT-ERROR PIC 9(6) VALUE 501003.
05 ERR-COM-ALLOC-ERROR PIC 9(6) VALUE 501004.
05 ERR-COM-ALLOC-RETRY PIC 9(6) VALUE 501005.
05 ERR-COM-CONN-ERROR PIC 9(6) VALUE 501006.
05 ERR-COM-SEND-ERROR PIC 9(6) VALUE 501008.
05 ERR-COM-RECV-RESP-ERR PIC 9(6) VALUE 501009.
05 ERR-COM-RESP-TYPE PIC 9(6) VALUE 501010.
05 ERR-COM-RESP-MSN PIC 9(6) VALUE 501011.
05 ERR-COM-RESP-FATAL PIC 9(6) VALUE 501012.
05 ERR-COM-MSG-ERROR PIC 9(6) VALUE 501013.
05 ERR-COM-BIG-INDIAN PIC 9(6) VALUE 501014.
05 ERR-COM-TSH-ERROR PIC 9(6) VALUE 501015.
05 ERR-COM-CCSID-ERROR PIC 9(6) VALUE 501016.
05 ERR-COM-MSH-ERROR PIC 9(6) VALUE 501017.
05 ERR-COM-MQX-ERROR PIC 9(6) VALUE 501018.
05 ERR-COM-INIT-ERROR PIC 9(6) VALUE 501019.
05 ERR-COM-FAP-ERROR PIC 9(6) VALUE 501020.
05 ERR-COM-MSG-SIZE PIC 9(6) VALUE 501021.
05 ERR-COM-WRAP-ERROR PIC 9(6) VALUE 501022.
05 ERR-COM-MCP-DOWN PIC 9(6) VALUE 501023.
05 ERR-COM-DOWN PIC 9(6) VALUE 501024.
05 ERR-COM-NOT-FOUND PIC 9(6) VALUE 501025.

```

```

05 ERR-COM-ERROR PIC 9(6) VALUE 501026.
05 ERR-COM-BUSY PIC 9(6) VALUE 501027.
05 ERR-COM-RESYNC-ERROR PIC 9(6) VALUE 501028.
05 ERR-COM-STATUS-ERROR PIC 9(6) VALUE 501029.
05 ERR-COM-LENGTH-ERROR PIC 9(6) VALUE 501030.
05 ERR-COM-MSG-PER-BATCH PIC 9(6) VALUE 501031.
05 ERR-COM-MAX-TRANSM-SIZE PIC 9(6) VALUE 501032.
05 ERR-COM-RESET-MSN PIC 9(6) VALUE 501050.

05 ERR-INT-LINK-ERROR PIC 9(6) VALUE 400000.
05 ERR-INT-LINK-COM-SIZE PIC 9(6) VALUE 400001.
05 ERR-INT-LINK-COM-DATA PIC 9(6) VALUE 400002.
05 ERR-INT-RETURN-ERROR PIC 9(6) VALUE 400003.
05 ERR-INT-MOVE-ERROR PIC 9(6) VALUE 400010.
05 ERR-INT-STRUC-MISSING PIC 9(6) VALUE 402000.
05 ERR-INT-STRUC-ERROR PIC 9(6) VALUE 402090.

05 ERR-LOGIC-NOT-SUPPORTED PIC 9(6) VALUE 300000.
05 ERR-LOGIC-STARTED-WRONG PIC 9(6) VALUE 300010.
05 ERR-LOGIC-REPEATED-FAILURE PIC 9(6) VALUE 300020.
05 ERR-LOGIC-LOCKS-EXCEEDED PIC 9(6) VALUE 300030.
05 ERR-LOGIC-MISSING-RECORD PIC 9(6) VALUE 301000.
05 ERR-LOGIC-RECORD-DUPLICATED PIC 9(6) VALUE 301010.
05 ERR-LOGIC-Q-CKP-MISSING PIC 9(6) VALUE 309010.

05 ERR-PROC-SYSTEM-STOPPED PIC 9(6) VALUE 100000.
05 ERR-PROC-SYSTEM-ACTIVE PIC 9(6) VALUE 100010.
05 ERR-PROC-SYS-START-NOQDR PIC 9(6) VALUE 100011.
05 ERR-PROC-SYS-START-MAXQDR PIC 9(6) VALUE 100012.
05 ERR-PROC-SYS-START-MAXCOM PIC 9(6) VALUE 100013.
05 ERR-PROC-SYS-START-NOSYS PIC 9(6) VALUE 100090.
05 ERR-PROC-Q-EXCEEDED-DEPTH PIC 9(6) VALUE 101000.
05 ERR-PROC-Q-CONCURRENT-UPD PIC 9(6) VALUE 101010.
05 ERR-PROC-Q-NOTFOUND PIC 9(6) VALUE 101015.
05 ERR-PROC-Q-STOPPED PIC 9(6) VALUE 101090.
05 ERR-PROC-Q-DISABLED PIC 9(6) VALUE 101091.
05 ERR-PROC-QSN-LIMIT-REACHED PIC 9(6) VALUE 102090.
05 ERR-PROC-FILE-SPACE-PUT PIC 9(6) VALUE 102091.
05 ERR-PROC-FILE-SPACE PIC 9(6) VALUE 102092.
05 ERR-PROC-DUAL-Q-ERROR PIC 9(6) VALUE 104021.
05 ERR-PROC-DUAL-Q-FILE PIC 9(6) VALUE 104022.
05 ERR-PROC-DUAL-Q-LOGIC PIC 9(6) VALUE 104023.
05 ERR-PROC-TRIGGER-ERROR PIC 9(6) VALUE 105090.
05 ERR-PROC-TRIGGER-DATA PIC 9(6) VALUE 105091.
05 ERR-PROC-NOT-AUTHORIZED PIC 9(6) VALUE 109000.

05 ERR-WARN-SYS-STARTED-W-ERR PIC 9(6) VALUE 010000.
05 ERR-WARN-SYS-STARTED-W-FILER PIC 9(6) VALUE 010001.
05 ERR-WARN-SYS-STARTED-W-COMER PIC 9(6) VALUE 010002.
05 ERR-WARN-SYS-STARTED-W-CHANG PIC 9(6) VALUE 010003.

05 ERR-WARN-COM-CONNECT PIC 9(6) VALUE 005000.
05 ERR-WARN-COM-OPENED PIC 9(6) VALUE 005001.
05 ERR-WARN-COM-QUEUE-OPENED PIC 9(6) VALUE 005002.
05 ERR-WARN-COM-LU62-CONNECT PIC 9(6) VALUE 005003.
05 ERR-WARN-COM-RECEIVER-ALLOCC PIC 9(6) VALUE 005004.
05 ERR-WARN-COM-QUEUE-EMPTY PIC 9(6) VALUE 005005.
05 ERR-WARN-COM-QUEUE-CLOSED PIC 9(6) VALUE 005006.
05 ERR-WARN-COM-DISC PIC 9(6) VALUE 005007.
05 ERR-WARN-COM-SHUT PIC 9(6) VALUE 005008.
05 ERR-WARN-COM-SHUT-SENT PIC 9(6) VALUE 005009.

05 ERR-FUNCTION-STARTED PIC 9(6) VALUE 000100.
05 ERR-FUNCTION-DONE PIC 9(6) VALUE 001000.
05 ERR-FUNCTION-NOT-DONE PIC 9(6) VALUE 001090.

05 ERR-WARN-SYS-STARTED PIC 9(6) VALUE 000000.

05 SYNCH-MSN-ERROR PIC 9(6) VALUE 3.
05 SYNCH-MSG-DUP PIC 9(6) VALUE 4.
05 LU62-FREE-ERROR PIC 9(6) VALUE 10.
05 LU62-EIB-ERROR PIC 9(6) VALUE 11.
05 LU62-STAT-ERROR PIC 9(6) VALUE 12.
05 LU62-ALLOC-ERROR PIC 9(6) VALUE 13.
05 LU62-ALLOC-RETRY-ERROR PIC 9(6) VALUE 14.
05 LU62-CONN-ERROR PIC 9(6) VALUE 15.
05 LU62-SEND-ERROR PIC 9(6) VALUE 16.
05 LU62-RECV-RESP-ERROR PIC 9(6) VALUE 17.
05 INVLD-RESP-TYPE PIC 9(6) VALUE 23.
05 INVLD-RESP-MSN PIC 9(6) VALUE 24.
05 FATAL-RESP-TYPE PIC 9(6) VALUE 25.
05 RECOVERABLE-RESP-TYPE PIC 9(6) VALUE 26.
05 PARSER-MSN-ERROR PIC 9(6) VALUE 29.

05 PARSER-TYPE-ERROR PIC 9(6) VALUE 30.
05 PARSER-PDM-ERROR PIC 9(6) VALUE 31.
05 PARSER-SID-ERROR PIC 9(6) VALUE 32.
05 PARSER-PN-ERROR PIC 9(6) VALUE 33.
05 PARSER-KEY-ERROR PIC 9(6) VALUE 34.
05 PARSER-APID-ERROR PIC 9(6) VALUE 35.
05 PARSER-ORG-DT-ERROR PIC 9(6) VALUE 38.
05 PARSER-ORIG-MSN-ERROR PIC 9(6) VALUE 39.
05 PARSER-BODY-ERROR PIC 9(6) VALUE 40.
05 PARSER-STATUS-ERROR PIC 9(6) VALUE 41.
05 PARSER-LENGTH-ERROR PIC 9(6) VALUE 42.
05 MCONN-ERROR PIC 9(6) VALUE 51.
05 MQOPEN-ERROR PIC 9(6) VALUE 52.
05 MQGET-ERROR PIC 9(6) VALUE 53.
05 MQPUT-ERROR PIC 9(6) VALUE 54.
05 MQPT1-ERROR PIC 9(6) VALUE 55.
05 MQCLOSE-ERROR PIC 9(6) VALUE 56.
05 MQDISC-ERROR PIC 9(6) VALUE 57.
05 QM-OTHER-ERROR PIC 9(6) VALUE 60.
05 RECV-RETURN-LON-STATUS PIC 9(6) VALUE 80.
05 RECV-RETURN-LON-TYPE PIC 9(6) VALUE 81.
05 SIDRC-RETURN-MLP-FORMAT PIC 9(6) VALUE 91.

*-----*
EJECT
*-----*
77 WS-DATA-LENGTH PIC S9(8) COMP VALUE ZERO.
77 WS-DATA-LENGTH PIC S9(4) COMP VALUE ZERO.
01 WS-DATA-ALL.
05 WS-DATA-WITH-QUEUE.
10 WS-DATA-WITH-TIMES.
12 WS-DATA-WITH-FUNCTION.
15 WS-TRANSID PIC X(5) VALUE 'TST2 '.
15 WS-DATA-FUNCTION PIC XXXX.
88 VALID-DATA-FUNCTION VALUE 'PUT'
'GET'
'BOTH'
'PUT1'
'INQ'
'PUTR'
'GETD'
'HELP'
'?''.
88 WS-HELP-FUN VALUE 'HELP' '?''.
88 WS-PUT VALUE 'PUT' '.
88 WS-GET VALUE 'GET' '.
88 WS-BOTH VALUE 'BOTH' '.
88 WS-PUT1 VALUE 'PUT1' '.
88 WS-PUT-WITH-REPLY VALUE 'PUTR' '.
88 WS-GET-WITH-DELETE VALUE 'GETD' '.
88 WS-INQ VALUE 'INQ' '.

12 FILLER PIC X VALUE ' '.
12 WS-DATA-TIMES PIC 99 VALUE 01.
10 WS-DATA-SYNC-FLAG PIC X VALUE ' '.
10 WS-DATA-QUEUE PIC X(48) VALUE SPACES.

01 MQI-SELECTOR-COUNT.
05 WS-SELECTOR-COUNT PIC S9(8) COMP VALUE ZERO.

01 MQI-SELECTOR.
05 MQI-SELECTOR-ENTRY OCCURS 40 TIMES
PIC S9(8) COMP.

01 MQI-IN-ATTR-COUNT.
05 WS-IN-ATTR-COUNT PIC S9(8) COMP VALUE +40.

01 MQI-IN-ATTR.
05 MQI-IN-ATTR-ENTRY OCCURS 40 TIMES
PIC S9(8) COMP.

*-----*
COPY MQWTRACE.
EJECT
*-----*
01 WS-PASSED-INFO.
COPY TTITST2.

```

```

EJECT
-----*
01 WS-NEED-REPLY.
05 FILLER PIC X(80) VALUE
'Please enter REPLY QUEUE name with trailing blanks or ErsEOF
- ' (eg. Ctrl - Del)'.
EJECT
-----*
01 WS-HELP.
05 FILLER PIC X(80) VALUE
'TST2 is a test facility for SENDING / RECEIVING messages'.
05 FILLER PIC X(80) VALUE
'The format of command is as follows:'.
05 FILLER PIC X(80) VALUE
'TST2 XXXX £NN' QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ
- 'QQ'.
05 FILLER PIC X(80) VALUE SPACES.
05 FILLER PIC X(80) VALUE
'(NOTE a single space or comma separates the params)'.
05 FILLER PIC X(80) VALUE
'XXXX 4-character function code, pad with trailing blank'.
05 FILLER PIC X(80) VALUE
'HELP - DISPLAY THIS HELP TEXT'.
05 FILLER PIC X(80) VALUE
'PUT - MQPUT MESSAGES'.
05 FILLER PIC X(80) VALUE
'PUT1 - MQPUT1 MESSAGES'.
05 FILLER PIC X(80) VALUE
'PUTR - MQPUT W/ REPLY MESSAGE'.
05 FILLER PIC X(80) VALUE
'GET - MQGET MESSAGES'.
05 FILLER PIC X(80) VALUE
'GETD - MQGET W/ BROWSE & DELETE'.
05 FILLER PIC X(80) VALUE
'BOTH - MQPUT FOLLOWED BY MQGET'.
05 FILLER PIC X(80) VALUE
'INQ - INQ ABOUT QUEUE (no count NN)'.
05 FILLER PIC X(80) VALUE
'NN 2-digit number with leading zero (01 TO 99)'.
05 FILLER PIC X(80) VALUE
'QQQQ A 48-character field giving the name of a queue.'.
05 FILLER PIC X(80) VALUE
'An additional prompt will ask for the name of the reply qu
- 'eue for PUTR option.'.
01 WS-HELP-RED REDEFINES WS-HELP.
05 WS-HELP-LINE OCCURS 17 TIMES
PIC X(80).
-----*
EJECT
-----*
01 WS-ALL-MSG.
05 WS-OK-MSG.
10 FILLER PIC X(80) VALUE
'FULL CYCLE HAS BEEN PERFORMED SUCCESSFULLY'.
10 WS-OK-MSG-1 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-2 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-3 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-4 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-5 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-6 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-7 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-8 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-9 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-10 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-11 PIC X(80) VALUE SPACES.
10 WS-OK-MSG-12 PIC X(80) VALUE SPACES.
05 WS-ERR-LINES.
10 FILLER PIC X(400) VALUE SPACES.

01 WS-OK-STATS-LINE-1.
05 FILLER PIC X(20) VALUE
'QUEUE USED -'.
05 WS-OK-QUEUE PIC X(48).

01 WS-OK-STATS-LINE-2.
05 FILLER PIC X(20) VALUE
'REPLY Q-'.
05 WS-OK-QUEUE-REPLY PIC X(48).

01 WS-OK-STATS-LINE-3.
05 FILLER PIC X(40) VALUE
'NUMBER OF MESSAGES PROCESSED -'.

```

```

05 WS-OK-MESSAGES PIC Z99.

01 WS-OK-STATS-LINE-4.
05 FILLER PIC X(40) VALUE
'TOTAL SECONDS .....-'.
05 WS-OK-TIME PIC X(8).

01 WS-INQ-DETS.
05 FILLER PIC X(6)
VALUE 'TYPE: '.
05 WS-I-TYPE PIC 99.
05 FILLER PIC X(10)
VALUE 'INHIBIT: '.
05 WS-I-INHIBIT PIC 99.
05 FILLER PIC X(7)
VALUE 'MAXL: '.
05 WS-I-MAXL PIC 999999.

-----*
EJECT
-----*
01 WS-ERROR-MESSAGES.
05 WS-ERR-DATA.
10 FILLER PIC X(13) VALUE
'DATA ERROR:'.
10 FILLER PIC X(9) VALUE
'LENGTH='.
10 WS-ERR-DATA-LENGTH PIC 9(8) VALUE ZERO.
10 FILLER PIC X(9) VALUE
', DATA ='.
10 WS-ERR-DATA-AREA PIC X(200) VALUE SPACES.
10 FILLER PIC X(4) VALUE
'****'.

05 WS-ERR-DISPLAY.
10 FILLER PIC X(13) VALUE
'MQ ERROR:'.
10 FILLER PIC X(9) VALUE
'LEVEL ='.
10 WS-LEVEL PIC X(8) VALUE SPACES.
10 FILLER PIC X(9) VALUE
', FUNC ='.
10 WS-FUNCTION PIC X(8) VALUE SPACES.
10 FILLER PIC X(9) VALUE
', CC ='.
10 WS-ERR-DISPLAY-CCODE PIC 9(4) VALUE ZERO.
10 FILLER PIC X(9) VALUE
', RC ='.
10 WS-ERR-DISPLAY-RCODE PIC 9(4) VALUE ZERO.
10 FILLER PIC X(4) VALUE
'****'.

EJECT
-----*
01 FILLER.
*****
** FILE NAME: CMQV **
** **
** DESCRIPTIVE NAME: COBOL copy file for MQI constants **
** **
** VERSION 2.1.0 **
** **
** FUNCTION: This file declares the constants **
** which form part of the IBM Message **
** Queue Interface (MQI). **
** **
*****

*****
** Values Related to MQDLH Structure **
*****
** Structure Identifier **
10 MQDLH-STRUC-ID PIC X(4) VALUE 'DLH '.

** Structure Version Number **
10 MQDLH-VERSION-1 PIC S9(9) BINARY VALUE 1.

*****
** Values Related to MQGMO Structure **
*****

```



```

** Structure Identifier
10 MQGMO-STRUC-ID PIC X(4) VALUE 'GMO '.

** Structure Version Number
10 MQGMO-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Get-Message Options
10 MQGMO-WAIT PIC S9(9) BINARY VALUE 1.
10 MQGMO-NO-WAIT PIC S9(9) BINARY VALUE 0.
10 MQGMO-BROWSE-FIRST PIC S9(9) BINARY VALUE 16.
10 MQGMO-BROWSE-NEXT PIC S9(9) BINARY VALUE 32.
10 MQGMO-ACCEPT-TRUNCATED-MSG PIC S9(9) BINARY VALUE 64.
10 MQGMO-SET-SIGNAL PIC S9(9) BINARY VALUE 8.
10 MQGMO-SYNCPPOINT PIC S9(9) BINARY VALUE 2.
10 MQGMO-NO-SYNCPPOINT PIC S9(9) BINARY VALUE 4.
10 MQGMO-MSG-UNDER-CURSORS PIC S9(9) BINARY VALUE 256.
10 MQGMO-LOCK PIC S9(9) BINARY VALUE 512.
10 MQGMO-UNLOCK PIC S9(9) BINARY VALUE 1024.

** Wait Interval
10 MQWI-UNLIMITED PIC S9(9) BINARY VALUE -1.

*****
** Values Related to MQMD Structure **
*****

** Structure Identifier
10 MQMD-STRUC-ID PIC X(4) VALUE 'MD '.

** Structure Version Number
10 MQMD-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Report Options
10 MQRO-NONE PIC S9(9) BINARY VALUE 0.

** Message Types
10 MQMT-REQUEST PIC S9(9) BINARY VALUE 1.
10 MQMT-REPLY PIC S9(9) BINARY VALUE 2.
10 MQMT-DATAGRAM PIC S9(9) BINARY VALUE 8.
10 MQMT-REPORT PIC S9(9) BINARY VALUE 4.

** Expiry Value
10 MQEI-UNLIMITED PIC S9(9) BINARY VALUE -1.

** Feedback Values
10 MQFB-NONE PIC S9(9) BINARY VALUE 0.
10 MQFB-QUIT PIC S9(9) BINARY VALUE 256.
10 MQFB-SYSTEM-FIRST PIC S9(9) BINARY VALUE 1.
10 MQFB-SYSTEM-LAST PIC S9(9) BINARY VALUE 65535.
10 MQFB-APPL-FIRST PIC S9(9) BINARY VALUE 65536.
10 MQFB-APPL-LAST PIC S9(9) BINARY VALUE 99999999.

* format
10 MQFMT-NONE PIC X(8) VALUE SPACES.
10 MQFMT-DEAD-LETTER-Q-HEADER PIC X(8) VALUE 'MQDLQH'.
10 MQFMT-TRIGGER PIC X(8) VALUE 'MQTRIG'.
10 MQFMT-XMIT-Q-HEADER PIC X(8) VALUE 'MQXMIT'.

** Encoding Value
10 MQENC-NATIVE PIC S9(9) BINARY VALUE 785.

** Encoding Masks
10 MQENC-INTEGGER-MASK PIC S9(9) BINARY VALUE 15.
10 MQENC-DECIMAL-MASK PIC S9(9) BINARY VALUE 240.
10 MQENC-FLOAT-MASK PIC S9(9) BINARY VALUE 3840.
10 MQENC-RESERVED-MASK PIC S9(9) BINARY VALUE -4096.

** Encodings for Binary Integers
10 MQENC-INTEGGER-UNDEFINED PIC S9(9) BINARY VALUE 0.
10 MQENC-INTEGGER-NORMAL PIC S9(9) BINARY VALUE 1.
10 MQENC-INTEGGER-REVERSED PIC S9(9) BINARY VALUE 2.

** Encodings for Packed-Decimal Integers
10 MQENC-DECIMAL-UNDEFINED PIC S9(9) BINARY VALUE 0.
10 MQENC-DECIMAL-NORMAL PIC S9(9) BINARY VALUE 16.
10 MQENC-DECIMAL-REVERSED PIC S9(9) BINARY VALUE 32.

** Encodings for Floating-Point Numbers
10 MQENC-FLOAT-UNDEFINED PIC S9(9) BINARY VALUE 0.
10 MQENC-FLOAT-IEEEE-NORMAL PIC S9(9) BINARY VALUE 256.
10 MQENC-FLOAT-IEEEE-REVERSED PIC S9(9) BINARY VALUE 512.
10 MQENC-FLOAT-S390 PIC S9(9) BINARY VALUE 768.

** Coded Character-Set Identifier
10 MQCCSI-Q-MGR PIC S9(9) BINARY VALUE 0.

** Persistence Values
10 MQPER-PERSISTENT PIC S9(9) BINARY VALUE 1.
10 MQPER-PERSISTENCE-AS-Q-DEF PIC S9(9) BINARY VALUE 2.

** Message Id Value
10 MQMI-NONE PIC X(24) VALUE LOW-VALUES.

** Correlation Id Value
10 MQCI-NONE PIC X(24) VALUE LOW-VALUES.

*****
** Values Related to MQOD Structure **
*****

** Structure Identifier
10 MQOD-STRUC-ID PIC X(4) VALUE 'OD '.

** Structure Version Number
10 MQOD-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Object Types
10 MQOT-Q PIC S9(9) BINARY VALUE 1.

*****
** Values Related to MQPMO Structure **
*****

** Structure Identifier
10 MQPMO-STRUC-ID PIC X(4) VALUE 'PMO '.

** Structure Version Number
10 MQPMO-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Put-Message Options
10 MQPMO-SYNCPPOINT PIC S9(9) BINARY VALUE 2.
10 MQPMO-NO-SYNCPPOINT PIC S9(9) BINARY VALUE 4.

*****
** Values Related to MQTM Structure **
*****

** Structure Identifier
10 MQTM-STRUC-ID PIC X(4) VALUE 'TM '.

** Structure Version Number
10 MQTM-VERSION-1 PIC S9(9) BINARY VALUE 1.

*****
** Values Related to MQCLOSE Call **
*****

** Close Options
10 MQCO-NONE PIC S9(9) BINARY VALUE 0.

*****
** Values Related to MQINQ Call **
*****

** Character-Attribute Selectors
10 MQCA-BASE-Q-NAME PIC S9(9) BINARY VALUE 2002.
10 MQCA-CREATION-DATE PIC S9(9) BINARY VALUE 2004.
10 MQCA-CREATION-TIME PIC S9(9) BINARY VALUE 2005.
10 MQCA-FIRST PIC S9(9) BINARY VALUE 2001.
10 MQCA-INITIATION-Q-NAME PIC S9(9) BINARY VALUE 2008.
10 MQCA-LAST PIC S9(9) BINARY VALUE 4000.
10 MQCA-PROCESS-NAME PIC S9(9) BINARY VALUE 2012.
10 MQCA-Q-DESC PIC S9(9) BINARY VALUE 2013.
10 MQCA-Q-NAME PIC S9(9) BINARY VALUE 2016.
10 MQCA-REMOTE-Q-MGR-NAME PIC S9(9) BINARY VALUE 2017.
10 MQCA-REMOTE-Q-NAME PIC S9(9) BINARY VALUE 2018.

** Integer-Attribute Selectors
10 MQIA-CURRENT-Q-DEPTH PIC S9(9) BINARY VALUE 3.
10 MQIA-DEF-PERSISTENCE PIC S9(9) BINARY VALUE 5.

```

```

10 MQIA-DEFINITION-TYPE PIC S9(9) BINARY VALUE 7.
10 MQIA-FIRST PIC S9(9) BINARY VALUE 1.
10 MQIA-INHIBIT-GET PIC S9(9) BINARY VALUE 9.
10 MQIA-INHIBIT-PUT PIC S9(9) BINARY VALUE 10.
10 MQIA-LAST PIC S9(9) BINARY VALUE 2000.
10 MQIA-MAX-MSG-LENGTH PIC S9(9) BINARY VALUE 13.
10 MQIA-MAX-Q-DEPTH PIC S9(9) BINARY VALUE 15.
10 MQIA-OPEN-INPUT-COUNT PIC S9(9) BINARY VALUE 17.
10 MQIA-OPEN-OUTPUT-COUNT PIC S9(9) BINARY VALUE 18.
10 MQIA-Q-TYPE PIC S9(9) BINARY VALUE 20.
10 MQIA-SHAREABILITY PIC S9(9) BINARY VALUE 23.
10 MQIA-TRIGGER-CONTROL PIC S9(9) BINARY VALUE 24.
10 MQIA-TRIGGER-TYPE PIC S9(9) BINARY VALUE 28.
10 MQIA-USAGE PIC S9(9) BINARY VALUE 12.

** Integer Attribute Value Denoting 'Not Applicable'
10 MQIAV-NOT-APPLICABLE PIC S9(9) BINARY VALUE -1.

*****
** Values Related to MQOPEN Call **
*****

** Open Options
10 MQOO-INPUT-SHARED PIC S9(9) BINARY VALUE 2.
10 MQOO-INPUT-EXCLUSIVE PIC S9(9) BINARY VALUE 4.
10 MQOO-BROWSE PIC S9(9) BINARY VALUE 8.
10 MQOO-OUTPUT PIC S9(9) BINARY VALUE 16.
10 MQOO-INQUIRE PIC S9(9) BINARY VALUE 32.

*****
** Values Related to All Calls **
*****

** String Lengths
10 MQ-CREATION-DATE-LENGTH PIC S9(9) BINARY VALUE 12.
10 MQ-CREATION-TIME-LENGTH PIC S9(9) BINARY VALUE 8.
10 MQ-PROCESS-APPL-ID-LENGTH PIC S9(9) BINARY VALUE 256.
10 MQ-PROCESS-DESC-LENGTH PIC S9(9) BINARY VALUE 64.
10 MQ-PROCESS-ENV-DATA-LENGTH PIC S9(9) BINARY VALUE 128.
10 MQ-PROCESS-NAME-LENGTH PIC S9(9) BINARY VALUE 48.
10 MQ-PROCESS-USER-DATA-LENGTH PIC S9(9) BINARY VALUE 128.
10 MQ-Q-DESC-LENGTH PIC S9(9) BINARY VALUE 64.
10 MQ-Q-NAME-LENGTH PIC S9(9) BINARY VALUE 48.
10 MQ-Q-MGR-DESC-LENGTH PIC S9(9) BINARY VALUE 64.
10 MQ-Q-MGR-NAME-LENGTH PIC S9(9) BINARY VALUE 48.
10 MQ-TRIGGER-DATA-LENGTH PIC S9(9) BINARY VALUE 64.

*****
** Completion Codes
10 MQCC-OK PIC S9(9) BINARY VALUE 0.
10 MQCC-WARNING PIC S9(9) BINARY VALUE 1.
10 MQCC-FAILED PIC S9(9) BINARY VALUE 2.

** Reason Codes
10 MQRC-NONE PIC S9(9) BINARY VALUE 0.
10 MQRC-ACCESS-RESTRICTED PIC S9(9) BINARY VALUE 2000.
10 MQRC-ALIAS-BASE-Q-TYPE-ERROR PIC S9(9) BINARY VALUE 2001.
10 MQRC-ALREADY-CONNECTED PIC S9(9) BINARY VALUE 2002.
10 MQRC-BUFFER-ERROR PIC S9(9) BINARY VALUE 2004.
10 MQRC-BUFFER-LENGTH-ERROR PIC S9(9) BINARY VALUE 2005.
10 MQRC-CHAR-ATTR-LENGTH-ERROR PIC S9(9) BINARY VALUE 2006.
10 MQRC-CHAR-ATTRS-ERROR PIC S9(9) BINARY VALUE 2007.
10 MQRC-CHAR-ATTRS-TOO-SHORT PIC S9(9) BINARY VALUE 2008.
10 MQRC-CONNECTION-BROKEN PIC S9(9) BINARY VALUE 2009.
10 MQRC-DATA-LENGTH-ERROR PIC S9(9) BINARY VALUE 2010.
10 MQRC-EXPIRY-ERROR PIC S9(9) BINARY VALUE 2013.
10 MQRC-FEEDBACK-ERROR PIC S9(9) BINARY VALUE 2014.
10 MQRC-GET-INHIBITED PIC S9(9) BINARY VALUE 2016.
10 MQRC-HANDLE-NOT-AVAILABLE PIC S9(9) BINARY VALUE 2017.
10 MQRC-HCONN-ERROR PIC S9(9) BINARY VALUE 2018.
10 MQRC-HOBJ-ERROR PIC S9(9) BINARY VALUE 2019.
10 MQRC-INT-ATTR-COUNT-ERROR PIC S9(9) BINARY VALUE 2021.
10 MQRC-INT-ATTR-COUNT-TOO-SMALL PIC S9(9) BINARY VALUE 2022.
10 MQRC-INT-ATTRS-ARRAY-ERROR PIC S9(9) BINARY VALUE 2023.
10 MQRC-MAX-CONNS-LIMIT-REACHED PIC S9(9) BINARY VALUE 2025.
10 MQRC-MD-ERROR PIC S9(9) BINARY VALUE 2026.
10 MQRC-MISSING-REPLY-TO-Q PIC S9(9) BINARY VALUE 2027.
10 MQRC-MSG-TYPE-ERROR PIC S9(9) BINARY VALUE 2029.
10 MQRC-MSG-TOO-BIG-FOR-Q PIC S9(9) BINARY VALUE 2030.
10 MQRC-NO-MSG-AVAILABLE PIC S9(9) BINARY VALUE 2033.
10 MQRC-NO-MSG-UNDER-CURSOR PIC S9(9) BINARY VALUE 2034.
10 MQRC-NOT-AUTHORIZED PIC S9(9) BINARY VALUE 2035.

10 MQRC-NOT-OPEN-FOR-BROWSE PIC S9(9) BINARY VALUE 2036.
10 MQRC-NOT-OPEN-FOR-INPUT PIC S9(9) BINARY VALUE 2037.
10 MQRC-NOT-OPEN-FOR-INQUIRE PIC S9(9) BINARY VALUE 2038.
10 MQRC-NOT-OPEN-FOR-OUTPUT PIC S9(9) BINARY VALUE 2039.
10 MQRC-OBJECT-CHANGED PIC S9(9) BINARY VALUE 2041.
10 MQRC-OBJECT-IN-USE PIC S9(9) BINARY VALUE 2042.
10 MQRC-OBJECT-TYPE-ERROR PIC S9(9) BINARY VALUE 2043.
10 MQRC-OD-ERROR PIC S9(9) BINARY VALUE 2044.
10 MQRC-OPTION-NOT-VALID-FOR-TYPE PIC S9(9) BINARY VALUE 2045.
10 MQRC-OPTIONS-ERROR PIC S9(9) BINARY VALUE 2046.
10 MQRC-PERSISTENCE-ERROR PIC S9(9) BINARY VALUE 2047.
10 MQRC-PRIORITY-EXCEEDS-MAXIMUM PIC S9(9) BINARY VALUE 2049.
10 MQRC-PRIORITY-ERROR PIC S9(9) BINARY VALUE 2050.
10 MQRC-PUT-INHIBITED PIC S9(9) BINARY VALUE 2051.
10 MQRC-Q-FULL PIC S9(9) BINARY VALUE 2053.
10 MQRC-Q-SPACE-NOT-AVAILABLE PIC S9(9) BINARY VALUE 2056.
10 MQRC-Q-MGR-NAME-ERROR PIC S9(9) BINARY VALUE 2058.
10 MQRC-Q-MGR-NOT-AVAILABLE PIC S9(9) BINARY VALUE 2059.
10 MQRC-REPORT-OPTIONS-ERROR PIC S9(9) BINARY VALUE 2061.
10 MQRC-SECURITY-ERROR PIC S9(9) BINARY VALUE 2063.
10 MQRC-SELECTOR-COUNT-ERROR PIC S9(9) BINARY VALUE 2065.
10 MQRC-SELECTOR-LIMIT-EXCEEDED PIC S9(9) BINARY VALUE 2066.
10 MQRC-SELECTOR-ERROR PIC S9(9) BINARY VALUE 2067.
10 MQRC-SELECTOR-NOT-FOR-TYPE PIC S9(9) BINARY VALUE 2068.
10 MQRC-SIGNAL-OUTSTANDING PIC S9(9) BINARY VALUE 2069.
10 MQRC-SIGNAL-REQUEST-ACCEPTED PIC S9(9) BINARY VALUE 2070.
10 MQRC-STORAGE-NOT-AVAILABLE PIC S9(9) BINARY VALUE 2071.
10 MQRC-SYNCPPOINT-NOT-AVAILABLE PIC S9(9) BINARY VALUE 2072.
10 MQRC-TRUNCATED-MSG-ACCEPTED PIC S9(9) BINARY VALUE 2079.
10 MQRC-TRUNCATED-MSG-FAILED PIC S9(9) BINARY VALUE 2080.
10 MQRC-UNEXPECTED-CONNECT-ERROR PIC S9(9) BINARY VALUE 2081.
10 MQRC-UNKNOWN-ALIAS-BASE-Q PIC S9(9) BINARY VALUE 2082.
10 MQRC-UNKNOWN-OBJECT-NAME PIC S9(9) BINARY VALUE 2085.
10 MQRC-UNKNOWN-OBJECT-Q-MGR PIC S9(9) BINARY VALUE 2086.
10 MQRC-UNKNOWN-REMOTE-Q-MGR PIC S9(9) BINARY VALUE 2087.
10 MQRC-WAIT-INTERVAL-ERROR PIC S9(9) BINARY VALUE 2090.
10 MQRC-XMIT-Q-TYPE-ERROR PIC S9(9) BINARY VALUE 2091.
10 MQRC-XMIT-Q-USAGE-ERROR PIC S9(9) BINARY VALUE 2092.
10 MQRC-PMO-ERROR PIC S9(9) BINARY VALUE 2173.
10 MQRC-GMO-ERROR PIC S9(9) BINARY VALUE 2186.

10 MQRC-UNEXPECTED-ERROR PIC S9(9) BINARY VALUE 2195.
10 MQRC-MSG-ID-ERROR PIC S9(9) BINARY VALUE 2206.
10 MQRC-CORREL-ID-ERROR PIC S9(9) BINARY VALUE 2207.

10 MQRC-FILE-SYSTEM-ERROR PIC S9(9) BINARY VALUE 2208.
10 MQRC-NO-MSG-LOCKED PIC S9(9) BINARY VALUE 2209.

*****
** Values Related to Queue Attributes **
*****

** Queue Types
10 MQQT-LOCAL PIC S9(9) BINARY VALUE 1.
10 MQQT-ALIAS PIC S9(9) BINARY VALUE 3.
10 MQQT-REMOTE PIC S9(9) BINARY VALUE 6.

** Queue Definition Types
10 MQQDT-PREDEFINED PIC S9(9) BINARY VALUE 1.

** Inhibit Get
10 MQQA-GET-INHIBITED PIC S9(9) BINARY VALUE 1.
10 MQQA-GET-ALLOWED PIC S9(9) BINARY VALUE 0.

** Inhibit Put
10 MQQA-PUT-INHIBITED PIC S9(9) BINARY VALUE 1.
10 MQQA-PUT-ALLOWED PIC S9(9) BINARY VALUE 0.

** Queue Shareability
10 MQQA-SHAREABLE PIC S9(9) BINARY VALUE 1.
10 MQQA-NOT-SHAREABLE PIC S9(9) BINARY VALUE 0.

** Message Delivery Sequence
10 MQMDS-FIFO PIC S9(9) BINARY VALUE 1.

** Trigger Control
10 MQTC-OFF PIC S9(9) BINARY VALUE 0.
10 MQTC-ON PIC S9(9) BINARY VALUE 1.

** Trigger Types
10 MQTT-NONE PIC S9(9) BINARY VALUE 0.

```

```

10 MQTT-FIRST PIC S9(9) BINARY VALUE 1.
10 MQTT-EVERY PIC S9(9) BINARY VALUE 2.

** Queue Usage
10 MQUS-NORMAL PIC S9(9) BINARY VALUE 0.
10 MQUS-TRANSMISSION PIC S9(9) BINARY VALUE 1.

*****
** Values Related to Process-Definition Attributes **
*****

** Application Type
10 MQAT-USER-FIRST PIC S9(9) BINARY VALUE 65536.
10 MQAT-USER-LAST PIC S9(9) BINARY VALUE 999999999.

*
10 MQAT-OS2 PIC S9(9) BINARY VALUE 4.
10 MQAT-DOS PIC S9(9) BINARY VALUE 5.
10 MQAT-AIX PIC S9(9) BINARY VALUE 6.
10 MQAT-OS400 PIC S9(9) BINARY VALUE 8.
10 MQAT-WINDOWS PIC S9(9) BINARY VALUE 9.
10 MQAT-CICS-VSE PIC S9(9) BINARY VALUE 10.
10 MQAT-VMS PIC S9(9) BINARY VALUE 12.
10 MQAT-GUARDIAN PIC S9(9) BINARY VALUE 13.
10 MQAT-VOS PIC S9(9) BINARY VALUE 14.

*****
** Values Related to Queue-Manager Attributes **
*****

** Syncpoint Availability
10 MQSP-AVAILABLE PIC S9(9) BINARY VALUE 1.

EJECT
*-----*
* ENVIRONMENT VALUES
*-----*
* - BEGIN - *** COPYBOOK: MQICENV *** - BEGIN - *
*-----*
* ENVIRONMENT VALUE - SYSTEM (ENV)
*-----*

02 ENV-DEFINITION.
03 ENV-DATA-FOR-SYSTEM.
05 ENV-PRODUCT-INSTALLED PIC X(4) VALUE 'MQM '.
88 ENV-PRODUCT-MQM VALUE 'MQM '.

05 ENV-PRODUCT-RUNTIME PIC X(4).
88 ENV-PRODUCT-RT-MQM VALUE 'MQM '.

05 ENV-LANG-INFO.
10 ENV-LANGUAGE-FILE-CODE PIC 99 VALUE 01.
10 ENV-LANGUAGE PIC X(24)
VALUE 'ENGLISH'.
05 ENV-DATE-FORMAT PIC 99 VALUE 01.
88 ENV-DATE-MDDYY VALUE 01.
88 ENV-DATE-YMMDD VALUE 02.
88 ENV-DATE-YYDDMM VALUE 03.
88 ENV-DATE-YYDDD VALUE 04.
88 ENV-DATE-DDMMYY VALUE 05.

03 ENV-DATA-FOR-TRAN.

05 ENV-MASTER-TERMINAL-TRAN.
10 ENV-MT-MASTER-TASK-ID PIC X(4) VALUE 'MQMT'.
10 ENV-MT-CONFIG-TASK-ID PIC X(4) VALUE 'MQMC'.
10 ENV-MT-MONITOR-TASK-ID PIC X(4) VALUE 'MQMM'.
10 ENV-MT-OPER-TASK-ID PIC X(4) VALUE 'MQMO'.
10 ENV-MT-DISP-TASK-ID PIC X(4) VALUE 'MQBQ'.
10 ENV-MT-QUEUE-TASK-ID PIC X(4) VALUE 'MQMQ'.
10 ENV-MT-QUEUEI-TASK-ID PIC X(4) VALUE 'MQDQ'.
10 ENV-MT-COM-TASK-ID PIC X(4) VALUE 'MQMH'.
10 ENV-MT-COMI-TASK-ID PIC X(4) VALUE 'MQDH'.
10 ENV-MT-SYS-TASK-ID PIC X(4) VALUE 'MQMS'.
10 ENV-MT-SYSI-TASK-ID PIC X(4) VALUE 'MQDS'.
10 ENV-MT-MONQ-TASK-ID PIC X(4) VALUE 'MQQM'.
10 ENV-MT-MONC-TASK-ID PIC X(4) VALUE 'MQCM'.
10 ENV-MT-SS-TASK-ID PIC X(4) VALUE 'MQMA'.

```

```

10 ENV-MT-SC-TASK-ID PIC X(4) VALUE 'MQMB'.
10 ENV-MT-SI-TASK-ID PIC X(4) VALUE 'MQMI'.
10 ENV-MT-SR-TASK-ID PIC X(4) VALUE 'MQMR'.
10 ENV-MT-SD-TASK-ID PIC X(4) VALUE 'MQMD'.
10 FILLER PIC X(4) VALUE SPACES.
10 FILLER PIC X(4) VALUE SPACES.
10 FILLER PIC X(4) VALUE SPACES.

05 ENV-INTERNAL-ITEMS-TRAN.
10 ENV-II-MONITOR PIC X(4) VALUE 'MQSM'.
10 ENV-II-M-RECOVERY PIC X(4) VALUE 'MQSR'.
10 ENV-II-Q-RECOVERY PIC X(4) VALUE 'MQSQ'.
10 ENV-II-START-STOP PIC X(4) VALUE 'MQSS'.
10 ENV-II-TRAN-AIP2 PIC X(4) VALUE 'MQ02'.
10 ENV-II-TRAN-COM-CHECKP PIC X(4) VALUE 'MQCP'.
10 ENV-II-TRAN-QUE-DELETE PIC X(4) VALUE 'MQDQ'.
10 ENV-II-TRAN-QUE-DEL-ALL PIC X(4) VALUE 'MQQA'.
10 FILLER PIC X(4) VALUE SPACES.
10 FILLER PIC X(4) VALUE SPACES.
10 FILLER PIC X(4) VALUE SPACES.

03 ENV-DATA-FOR-PROGRAMS.

05 ENV-MASTER-TERMINAL-PROGRAMS.
10 ENV-MT-MASTER-PROGRAM PIC X(8) VALUE 'MQMTP'.
10 ENV-MT-CONFIG-PROGRAM PIC X(8) VALUE 'MQMCFG'.
10 ENV-MT-MONITOR-PROGRAM PIC X(8) VALUE 'MQMMON'.
10 ENV-MT-OPER-PROGRAM PIC X(8) VALUE 'MQMOPR'.
10 ENV-MT-DISP-PROGRAM PIC X(8) VALUE 'MQPDISP'.
10 ENV-MT-QUEUE-PROGRAM PIC X(8) VALUE 'MQPMQUE'.
10 ENV-MT-QUEUEI-PROGRAM PIC X(8) VALUE 'MQPMQUE'.
10 ENV-MT-COM-PROGRAM PIC X(8) VALUE 'MQPMCOM'.
10 ENV-MT-COMI-PROGRAM PIC X(8) VALUE 'MQPMCOM'.
10 ENV-MT-SYS-PROGRAM PIC X(8) VALUE 'MQPMSYS'.
10 ENV-MT-SYSI-PROGRAM PIC X(8) VALUE 'MQPMSYS'.
10 ENV-MT-MONQ-PROGRAM PIC X(8) VALUE 'MQPMMOQ'.
10 ENV-MT-MONC-PROGRAM PIC X(8) VALUE 'MQPMMOC'.
10 ENV-MT-SS-PROGRAM PIC X(8) VALUE 'MQPMSS'.
10 ENV-MT-SC-PROGRAM PIC X(8) VALUE 'MQPMSC'.
10 ENV-MT-SI-PROGRAM PIC X(8) VALUE 'MQPMSI'.
10 ENV-MT-SR-PROGRAM PIC X(8) VALUE 'MQPMMSN'.
10 ENV-MT-SD-PROGRAM PIC X(8) VALUE 'MQPMDL'.
10 ENV-MT-CMD-PROGRAM PIC X(8) VALUE 'MQPCMD'.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.

05 ENV-INTERNAL-ITEMS-PROGRAMS.
10 ENV-II-LINK-ERROR PIC X(8) VALUE 'MQPERR '.
10 ENV-II-LINK-EIB1 PIC X(8) VALUE 'MQPEIB1 '.
10 ENV-II-LINK-AIP0 PIC X(8) VALUE 'MQPAIP0 '.
10 ENV-II-LINK-AIP1 PIC X(8) VALUE 'MQPAIP1 '.
10 ENV-II-LINK-AIP2 PIC X(8) VALUE 'MQPAIP2 '.

10 ENV-II-LINK-ECHO PIC X(8) VALUE 'MQPECHO '.
10 ENV-II-LINK-FINDQ PIC X(8) VALUE 'MQPFINDQ '.
10 ENV-II-LINK-QUE1 PIC X(8) VALUE 'MQPQUE1 '.
10 ENV-II-LINK-QUE2 PIC X(8) VALUE 'MQPQUE2 '.
10 ENV-II-LINK-INIT1 PIC X(8) VALUE 'MQPINIT1 '.
10 ENV-II-LINK-INIT2 PIC X(8) VALUE 'MQPINIT2 '.
10 ENV-II-LINK-SSQ PIC X(8) VALUE 'MQPSSQ '.
10 ENV-II-LINK-SCHK PIC X(8) VALUE 'MQPSCHK '.
10 ENV-II-LINK-SREC PIC X(8) VALUE 'MQPSREC '.
10 ENV-II-LINK-QRECOVERY PIC X(8) VALUE 'MQPQREC '.
10 ENV-II-LINK-SENDER PIC X(8) VALUE 'MQPSEND '.
10 ENV-II-LINK-RECIEVER PIC X(8) VALUE 'MQPRECV '.
10 ENV-II-LINK-COM-CHECKP PIC X(8) VALUE 'MQPCCKPT '.
10 ENV-II-LINK-QUE-DELETE PIC X(8) VALUE 'MQPQDEL '.
10 ENV-II-LINK-SET-MAP PIC X(8) VALUE 'MQPSMAP '.
10 ENV-II-LINK-LU21 PIC X(8) VALUE 'MQPLU21 '.
10 ENV-II-LINK-LU33 PIC X(8) VALUE 'MQPLU33 '.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.

03 ENV-DATA-FOR-MAPS.

05 ENV-MASTER-TERMINAL-MAPS.
10 ENV-MT-MASTER-MAPSCREEN PIC X(8) VALUE 'MQMTP'.
10 ENV-MT-CONFIG-MAPSCREEN PIC X(8) VALUE 'MQMCFG'.
10 ENV-MT-MONITOR-MAPSCREEN PIC X(8) VALUE 'MQMMON'.
10 ENV-MT-OPER-MAPSCREEN PIC X(8) VALUE 'MQMOPR'.

```

```

10 ENV-MT-DISP-MAPSCREEN PIC X(8) VALUE 'MQMDISP'.
10 ENV-MT-QUEUE-MAPSCREEN PIC X(8) VALUE 'MQMMQUE'.
10 ENV-MT-QUEUE1-MAPSCREEN PIC X(8) VALUE 'MQMMQUE'.
10 ENV-MT-COM-MAPSCREEN PIC X(8) VALUE 'MQMMCOM'.
10 ENV-MT-COM1-MAPSCREEN PIC X(8) VALUE 'MQMMCOM'.
10 ENV-MT-SYS-MAPSCREEN PIC X(8) VALUE 'MQMMSYS'.
10 ENV-MT-SYS1-MAPSCREEN PIC X(8) VALUE 'MQMMSYS'.
10 ENV-MT-MONQ-MAPSCREEN PIC X(8) VALUE 'MQMMMOQ'.
10 ENV-MT-MONC-MAPSCREEN PIC X(8) VALUE 'MQMMOC'.
10 ENV-MT-SS-MAPSCREEN PIC X(8) VALUE 'MQMMSS'.
10 ENV-MT-SC-MAPSCREEN PIC X(8) VALUE 'MQMMSC'.
10 ENV-MT-SI-MAPSCREEN PIC X(8) VALUE 'MQMMSI'.
10 ENV-MT-SR-MAPSCREEN PIC X(8) VALUE 'MQMMMSN'.
10 ENV-MT-SD-MAPSCREEN PIC X(8) VALUE 'MQMMDL'.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.

03 ENV-DATA-FOR-CONSTANTS.

05 ENV-CONFIG-DDNAME PIC X(8) VALUE 'MQFCNFG'.
05 ENV-SYSTEM-NUMBER PIC 9(4) VALUE 1.
05 ENV-MASTER-TERMINAL-CONS.
10 ENV-MT-TITLE PIC X(40) VALUE
' IBM MQSeries for VSE/ESA Version 1 '.

05 ENV-INTERNAL-ITEMS-CONS.
10 ENV-II-ERROR-TD PIC X(4) VALUE 'MQER'.
10 ENV-II-ERROR-CSMT PIC X(4) VALUE 'CSMT'.
10 ENV-II-SYSTEM-ANCHOR PIC X(8) VALUE 'MQTAQM'.
10 ENV-II-SYSTEM-PREFIX PIC X(4) VALUE 'MQI '.
10 ENV-II-DUMPCODE PIC X(4) VALUE 'MQ??'.
10 ENV-II-ENQ-INIT1 PIC X(8) VALUE 'MQPINIT1'.
10 ENV-II-SYSTEM-ENVIR PIC X(8) VALUE 'MQTENV '.
10 ENV-IT-UN-INIT-MSG PIC X(80) VALUE
'MQ90000: MQSERIES VSE ENVIRONMENT not initialized.'.
10 FILLER PIC X(80) VALUE SPACES.

-----*
* - END - *** COPYBOOK: MQICENV *** - END - *
-----*

EJECT
-----*
* COMMON PARMS
01 FILLER PIC X(8) VALUE 'PARMS:--'.
01 WS-HCONN-ADDR-AREA.
05 WS-HCONN-VALUE USAGE POINTER.

01 WS-HOBJ-ADDR-AREA.
05 WS-HOBJ-VALUE USAGE POINTER.

01 WS-HOBJ-ADDR-AREA-REPLY.
05 WS-HOBJ-VALUE-REPLY USAGE POINTER.

01 WS-CCODE-ADDR-AREA.
05 WS-CCODE-VALUE PIC S9(8) COMP.

01 WS-RCODE-ADDR-AREA.
05 WS-RCODE-VALUE PIC S9(8) COMP.

-----*
*--CONNECT PARM
01 WS-QM-NAME-AREA.
05 WS-QM-NAME-CONNECT PIC X(48).

*--OPEN PARM
01 WS-Q-NAME-AREA.
*****
** FILE NAME: CMQODV **
** DESCRIPTIVE NAME: COBOL copy file for MQOD structure **
** VERSION 2.1.0 **
** FUNCTION: This file declares the MQOD structure, **
** which forms part of the IBM Message **
** Queue Interface (MQI). **
*****

** MQOD structure
10 MQOD.
** Structure identifier
15 MQOD-STRUCID PIC X(4) VALUE 'OD '.
** Structure version number
15 MQOD-VERSION PIC S9(9) BINARY VALUE 1.
** Object type
15 MQOD-OBJECTTYPE PIC S9(9) BINARY VALUE 1.
** Object name
15 MQOD-OBJECTNAME PIC X(48) VALUE SPACES.
** Object queue manager name
15 MQOD-OBJECTQMGRNAME PIC X(48) VALUE SPACES.
** Dynamic queue name
15 MQOD-DYNAMICQNAME PIC X(48) VALUE '*'.
** Alternate user identifier
15 MQOD-ALTERNATEUSERID PIC X(12) VALUE SPACES.

01 WS-Q-OPEN-OPTIONS.
05 WS-Q-OPEN-OPTIONS-VALUE PIC S9(8) COMP.
EJECT

*--INQ
01 MQI-CHAR-ATTR-LENGTH.
05 WS-CHAR-ATTR-LENGTH PIC S9(8) COMP.
01 MQI-CHAR-ATTR.
05 WS-CHAR-ATTR PIC X(500) VALUE SPACES.
01 WS-D-FILLER REDEFINES MQI-CHAR-ATTR.
05 WS-A-DESC PIC X(64).
05 WS-A-Q-N PIC X(64).

*--PUT/GET PARM
01 WS-MSG-DESCRIPTOR.
*****
** FILE NAME: CMQMDV **
** DESCRIPTIVE NAME: COBOL copy file for MQMD structure **
** VERSION 2.1.0 **
** FUNCTION: This file declares the MQMD structure, **
** which forms part of the IBM Message **
** Queue Interface (MQI). **
*****

** MQMD structure
10 MQMD.
** Structure identifier
15 MQMD-STRUCID PIC X(4) VALUE 'MD '.
** Structure version number
15 MQMD-VERSION PIC S9(9) BINARY VALUE 1.
** Reserved
15 MQMD-REPORT PIC S9(9) BINARY VALUE 0.
** Message type
15 MQMD-MSGTYPE PIC S9(9) BINARY VALUE 8.
** Reserved
15 MQMD-EXPIRY PIC S9(9) BINARY VALUE -1.
** Feedback code
15 MQMD-FEEDBACK PIC S9(9) BINARY VALUE 0.
** Data encoding
15 MQMD-ENCODING PIC S9(9) BINARY VALUE 785.
** Coded character set identifier
15 MQMD-CODEDCHARSETID PIC S9(9) BINARY VALUE 0.
** Format name
15 MQMD-FORMAT PIC X(8) VALUE SPACES.
** Reserved
15 MQMD-PRIORITY PIC S9(9) BINARY VALUE 0.
** Message persistence
15 MQMD-PERSISTENCE PIC S9(9) BINARY VALUE 2.
** Message identifier
15 MQMD-MSGID PIC X(24) VALUE LOW-VALUES.
** Correlation identifier
15 MQMD-CORRELID PIC X(24) VALUE LOW-VALUES.
** Reserved
15 MQMD-BACKOUTCOUNT PIC S9(9) BINARY VALUE 0.
** Name of reply queue
15 MQMD-REPLYTOQ PIC X(48) VALUE SPACES.

```

```

** Name of reply queue manager
15 MQMD-REPLYTOQMGR PIC X(48) VALUE SPACES.
** Reserved
15 MQMD-USERIDENTIFIER PIC X(12) VALUE SPACES.
** Reserved
15 MQMD-ACCOUNTINGTOKEN PIC X(32) VALUE LOW-VALUES.
** Reserved
15 MQMD-APPLIDENTITYDATA PIC X(32) VALUE SPACES.
** Reserved
15 MQMD-PUTAPPLTYPE PIC S9(9) BINARY VALUE 0.
** Reserved
15 MQMD-PUTAPPLNAME PIC X(28) VALUE SPACES.
** Reserved
15 MQMD-PUTDATE PIC X(8) VALUE SPACES.
** Reserved
15 MQMD-PUTTIME PIC X(8) VALUE SPACES.
** Reserved
15 MQMD-APPLORIGINDATA PIC X(4) VALUE SPACES.

01 WS-PUT-OPTIONS.
*****
**
** FILE NAME: CMQPMOV
**
** DESCRIPTIVE NAME: COBOL copy file for MQPMO structure
**
** VERSION 2.1.0
**
** FUNCTION: This file declares the MQPMO structure,
** which forms part of the IBM Message
** Queue Interface (MQI).
**
*****

** MQPMO structure
10 MQPMO.
** Structure identifier
15 MQPMO-STRUCID PIC X(4) VALUE 'PMO '.
** Structure version number
15 MQPMO-VERSION PIC S9(9) BINARY VALUE 1.
** Reserved
15 MQPMO-OPTIONS PIC S9(9) BINARY VALUE 0.
** Reserved
15 MQPMO-TIMEOUT PIC S9(9) BINARY VALUE -1.
** Reserved
15 MQPMO-CONTEXT PIC S9(9) BINARY VALUE 0.
** Reserved
15 MQPMO-KNOWNDSTCOUNT PIC S9(9) BINARY VALUE 0.
** Reserved
15 MQPMO-UNKNOWNDESTCOUNT PIC S9(9) BINARY VALUE 0.
** Reserved
15 MQPMO-INVALIDDESTCOUNT PIC S9(9) BINARY VALUE 0.
** Resolved name of destination queue
15 MQPMO-RESOLVEDQNAME PIC X(48) VALUE SPACES.
** Resolved name of destination queue manager
15 MQPMO-RESOLVEDQMGRNAME PIC X(48) VALUE SPACES.

01 WS-GET-OPTIONS.
*****
**
** FILE NAME: CMQGMOV
**
** DESCRIPTIVE NAME: COBOL copy file for MQGMO structure
**
** VERSION 2.1.0
**
** FUNCTION: This file declares the MQGMO structure,
** which forms part of the IBM Message
** Queue Interface (MQI).
**
*****

** MQGMO structure
10 MQGMO.
** Structure identifier
15 MQGMO-STRUCID PIC X(4) VALUE 'GMO '.
** Structure version number
15 MQGMO-VERSION PIC S9(9) BINARY VALUE 1.
** Options
15 MQGMO-OPTIONS PIC S9(9) BINARY VALUE 0.
** Wait interval
15 MQGMO-WAITINTERVAL PIC S9(9) BINARY VALUE 0.
** Signal
15 MQGMO-SIGNAL1 PIC S9(9) BINARY VALUE 0.
** Reserved
15 MQGMO-SIGNAL2 PIC S9(9) BINARY VALUE 0.
** Resolved name of destination queue
15 MQGMO-RESOLVEDQNAME PIC X(48) VALUE SPACES.

01 WS-DATA-L-AREA.
05 WS-DATA-LENGTH-USER PIC S9(8) COMP VALUE +200.

01 WS-BUFFER-L-AREA.
05 WS-BUFFER-LENGTH PIC S9(8) COMP VALUE +200.

77 WS-MSG-LENGTH PIC S9(8) COMP VALUE +200.
01 WS-RECEIVE-BUFFER PIC X(1024) VALUE SPACE.
01 WS-REMAINDER PIC X(1024) VALUE SPACES.
01 WS-POINTER PIC 9(4).
01 WS-MSG-AREA.
05 FILLER PIC X(500) VALUE
'THIS IS A MESSAGE TEXT'.

01 WS-BUFFER-AREA.
05 WS-BUFFER-TS PIC X(16) VALUE SPACES.
05 WS-BUFFER-TEXT PIC X(500) VALUE SPACES.

COPY MQWEOWS.
-----*
LINKAGE SECTION.
-----*

01 LK-DATA.
05 FILLER PIC X.
EJECT
-----*
PROCEDURE DIVISION.
-----*

0000-MAIN-LINE.

---INITIALIZE
MOVE 'INIT ' TO WS-LEVEL.
PERFORM 1000-INITIALIZE
THRU 1000-EXIT.

-----*
PERFORM 1 TIMES

---SEND QUEUE RECORDS
IF WS-PUT OR WS-BOTH
THEN
PERFORM 2000-PUT-MESSAGES
THRU 2000-EXIT
END-IF

---GET QUEUE RECORDS
IF WS-GET OR WS-BOTH
THEN
PERFORM 3000-GET-MESSAGES
THRU 3000-EXIT
END-IF

IF WS-PUT1
THEN
PERFORM 4000-PUT1-MESSAGES
THRU 4000-EXIT
END-IF

IF WS-GET-WITH-DELETE
THEN
PERFORM 5000-GETD-MESSAGES
THRU 5000-EXIT
END-IF

IF WS-PUT-WITH-REPLY
THEN
PERFORM 6000-PUT-WITH-REPLY
THRU 6000-EXIT
END-IF
IF WS-INQ
THEN
PERFORM 6500-INQ-MESSAGES

```

TTPST2.Z

```

        THRU 6500-EXIT
    END-IF

    END-PERFORM.
-----*
0000-SEND-TOTALS.
    IF NOT WS-STARTED
        THEN
            PERFORM 7000-SEND-TOTALS.

*-----*
0000-RETURN.
    EXEC CICS RETURN
        END-EXEC.

    GOBACK.
    EJECT
-----*
1000-INITIALIZE.
-----*
* PURPOSE: SETUP DATA AREAS
-----*
*--GET STARTED CICS CODE
    EXEC CICS ASSIGN
        STARTCODE (WS-STARTCODE)
        END-EXEC.

*
*--SET TIME/DATE
    EXEC CICS ASKTIME
        ABSTIME(WS-ABSTIME)
        END-EXEC.

*
    EXEC CICS FORMATTIME
        ABSTIME(WS-ABSTIME)
        YYMMDD (WS-DATE-YYMMDD)
        END-EXEC.

    IF WS-DATE-YY > 50
        THEN
            MOVE 19          TO WS-DATE-CC
        ELSE
            MOVE 20          TO WS-DATE-CC.

*
    MOVE EIBTIME TO WS-TIME-9.
    EXEC CICS FORMATTIME
        ABSTIME (WS-ABSTIME)
        MMDDYY (WS-FORMATTED-DATE)
        DATESEP ('/')
        TIME (WS-FORMATTED-TIME)
        TIMESEP (':')
        END-EXEC.

*
*--GET INPUT INFO...
    IF START-WITH-DATA
        THEN
            PERFORM 1100-PASSED-INFO
        ELSE
            PERFORM 1200-SETUP-INPUT.

*--SET COMMON ERROR INFO
    MOVE ZERO TO ERR-CODE.
    MOVE 'TTPST2' TO ERR-PROGRAM.

*
-----*
1000-EXIT.
    EXIT.

    EJECT
-----*
1100-PASSED-INFO.
-----*
* PURPOSE: SETUP PASSED DATA AREAS
-----*
*--GET PASSED DATA
    MOVE LENGTH OF WS-PASSED-INFO TO WS-PASS-MSG-LENGTH.
    EXEC CICS RETRIEVE
        INTO (WS-PASSED-INFO)
        LENGTH (WS-PASS-MSG-LENGTH)
        END-EXEC.

        IF WS-PASS-MSG-LENGTH < LENGTH OF WS-PASSED-INFO
            THEN
                GO TO 0000-RETURN.

*
    SET WS-STARTED TO TRUE.
    MOVE TST2-FUNCTION TO WS-DATA-FUNCTION.
    MOVE TST2-PUT-NUM-MSG TO WS-PROCESS-TIMES.
    MOVE TST2-PUT-QUEUE-NAME TO WS-DATA-QUEUE.
    MOVE TST2-PUT-MSG-SIZE TO WS-MSG-LENGTH.
    MOVE TST2-PUT-MSG TO WS-MSG-AREA.
    MOVE TST2-PUT-MSG-TIMESTAMP TO WS-TIMESTAMP.

-----*
    EJECT
-----*
1100-SEND-HELP.
-----*
*--SEND HELPLIST
    EXEC CICS SEND
        FROM (WS-HELP)
        LENGTH (LENGTH OF WS-HELP)
        ERASE
        END-EXEC.

-----*
    EJECT
-----*
1200-SETUP-INPUT.
-----*
*--GET DATA
    MOVE LENGTH OF WS-RECEIVE-BUFFER TO WS-DATA-LENGTH
    EXEC CICS RECEIVE
        INTO(WS-RECEIVE-BUFFER)
        LENGTH(WS-DATA-LENGTH)

        END-EXEC

*-- Handle variable number of parameters.
    MOVE +1 TO WS-POINTER
    UNSTRING WS-RECEIVE-BUFFER DELIMITED BY ALL SPACES INTO
        WS-TRANSID
        WS-DATA-FUNCTION
        WITH POINTER WS-POINTER
    END-UNSTRING

*-- Handle mixed case.
    INSPECT WS-DATA-FUNCTION CONVERTING
        'abcdefghijklmnopqrstuvwxyz' TO
        'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

*-- Check for invalid function codes.
    IF (NOT VALID-DATA-FUNCTION) OR WS-HELP-FUN
        THEN
            PERFORM 1100-SEND-HELP
            GO TO 0000-RETURN
        END-IF

*--Extract the remaining options according to the function code.
    IF NOT WS-INQ THEN
        UNSTRING WS-RECEIVE-BUFFER DELIMITED BY ALL SPACES INTO
            WS-DATA-TIMES
            WS-DATA-QUEUE
            WS-TIMESTAMP
            WITH POINTER WS-POINTER
        END-UNSTRING
    ELSE
        UNSTRING WS-RECEIVE-BUFFER DELIMITED BY ALL SPACES INTO
            WS-DATA-QUEUE
            WITH POINTER WS-POINTER
        END-UNSTRING
    END-IF

*--CHECK WHAT WE'RE DOING
*-- --COMMAND IS "TST2 GET 01 QUEUENAME"
*
    MOVE WS-DATA-TIMES TO WS-PROCESS-TIMES.
    IF WS-PROCESS-TIMES EQUAL ZERO
        THEN
            MOVE 100 TO WS-PROCESS-TIMES.

*
*--IF REPLY ..SEND AND GET
    IF NOT WS-PUT-WITH-REPLY
        THEN
            GO TO 1000-EXIT.

*
*--IF REPLY ..SEND AND GET

```

```

EXEC CICS SEND
    FROM (WS-NEED-REPLY)
    LENGTH (LENGTH OF WS-NEED-REPLY)
    ERASE
END-EXEC.

EXEC CICS RECEIVE
    SET (ADDRESS OF LK-DATA)
    LENGTH(WS-DATA-LENGTH)
END-EXEC.

IF WS-DATA-LENGTH > 48
    THEN
        MOVE +48 TO WS-DATA-LENGTH.

MOVE WS-DATA-LENGTH TO WS-DATA-FLENGTH
MOVE LK-DATA(1:WS-DATA-FLENGTH) TO
    WS-REPLY-Q(1:WS-DATA-FLENGTH)

*
*-----*
EJECT
*-----*
2000-PUT-MESSAGES.
*-----*
* PURPOSE: CONNECT , OPEN
*     PUT
*     CLOSE,   DISCONNECT
*-----*
*
*--MQCONNECT TO QM
MOVE 'CONNECT' TO WS-FUNCTION.
MOVE SPACES TO WS-QM-NAME-CONNECT.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HCONN-VALUE TO NULL.
CALL 'MQCONN' USING WS-QM-NAME-AREA
    WS-HCONN-ADDR-AREA
    WS-CCODE-ADDR-AREA
    WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
    THEN
        GO TO 9900-ERR-DISPLAY.

*
*--MQOPEN QUEUE TO QM
MOVE 'OPEN' TO WS-FUNCTION.
MOVE MQOO-OUTPUT TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE SPACES TO MQOD-OBJECTQMGRNAME.
MOVE WS-DATA-QUEUE TO MQOD-OBJECTNAME.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HOBJ-VALUE TO NULL.
CALL 'MQOPEN' USING WS-HCONN-ADDR-AREA
    WS-Q-NAME-AREA
    WS-Q-OPEN-OPTIONS
    WS-HOBJ-ADDR-AREA
    WS-CCODE-ADDR-AREA
    WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
    THEN
        GO TO 9900-ERR-DISPLAY.

*-----*
PERFORM WS-PROCESS-TIMES TIMES

*--CHECK IF MUST PUT TIME STAMP ON MESSAGE
IF WS-PUT-TIMESTAMP
    THEN
        PERFORM 8000-GET-TIME-STAMP
        MOVE WS-COUNT TO WS-TIMESTAMP-COUNT
        MOVE WS-TIMESTAMP-VALUE TO WS-BUFFER-TS
        MOVE LENGTH OF WS-BUFFER-TS
            TO WS-BUFFER-LENGTH
        ADD WS-MSG-LENGTH TO WS-BUFFER-LENGTH
        MOVE WS-MSG-AREA TO WS-BUFFER-TEXT
    ELSE
        MOVE WS-MSG-LENGTH TO WS-BUFFER-LENGTH
        MOVE WS-MSG-AREA TO WS-BUFFER-AREA
    END-IF

*--MQPUT TO QUEUE TO QM
MOVE 'PUT' TO WS-FUNCTION
MOVE MQCC-OK TO WS-CCODE-VALUE
MOVE MQRC-NONE TO WS-RCODE-VALUE
CALL 'MQPUT' USING WS-HCONN-ADDR-AREA
    WS-HOBJ-ADDR-AREA
    WS-MSG-DESCRIPTOR
    WS-PUT-OPTIONS
    WS-BUFFER-L-AREA
    WS-BUFFER-AREA
    WS-CCODE-ADDR-AREA
    WS-RCODE-ADDR-AREA

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
    THEN
        GO TO 9900-ERR-DISPLAY
END-IF
ADD +1 TO WS-COUNT

*--SYNPOINT PUT SO ECHO CAN GET IT
*-- --CHECK IF "NEGATIVE " PROCESSING OPTION SPECIFIED
*     IF WS-DATA-SYNC-FLAG NOT EQUAL '-'
*     THEN
*         EXEC CICS SYNCPOINT
*         END-EXEC
*     END-IF

*-----*
END-PERFORM.
*-----*
*
*--MQCLOSE QUEUE TO QM
MOVE 'CLOSE' TO WS-FUNCTION.
MOVE ZERO TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQCLOSE' USING WS-HCONN-ADDR-AREA
    WS-HOBJ-ADDR-AREA
    WS-Q-OPEN-OPTIONS
    WS-CCODE-ADDR-AREA
    WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
    THEN
        GO TO 9900-ERR-DISPLAY.

*
*--MQDISC FROM QM
MOVE 'DISCONN' TO WS-FUNCTION.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQDISC' USING
    WS-HCONN-ADDR-AREA
    WS-CCODE-ADDR-AREA
    WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
    THEN
        GO TO 9900-ERR-DISPLAY.

*-----*
2000-EXIT.
EXIT.
EJECT
*-----*
3000-GET-MESSAGES.
*-----*
* PURPOSE: CONNECT , OPEN
*     GET
*     CLOSE,   DISCONNECT
*-----*
*
*--MQCONNECT TO QM
MOVE 'CONNECT' TO WS-FUNCTION.
MOVE SPACES TO WS-QM-NAME.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HCONN-VALUE TO NULL.
CALL 'MQCONN' USING WS-QM-NAME-AREA
    WS-HCONN-ADDR-AREA
    WS-CCODE-ADDR-AREA
    WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO

```

```

THEN
    GO TO 9900-ERR-DISPLAY.
*
*--MQOPEN  QUEUE TO QM
MOVE 'OPEN'  TO WS-FUNCTION.
MOVE MQOO-INPUT-SHARED TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE SPACES TO MQOD-OBJECTQMGRNAME.
MOVE WS-DATA-QUEUE TO MQOD-OBJECTNAME.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HOBJ-VALUE TO NULL.
CALL 'MQOPEN' USING WS-HCONN-ADDR-AREA
                   WS-Q-NAME-AREA
                   WS-Q-OPEN-OPTIONS
                   WS-HOBJ-ADDR-AREA
                   WS-CCODE-ADDR-AREA
                   WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.
*
-----*
PERFORM WS-PROCESS-TIMES TIMES
*
*--MQGET TO QUEUE TO QM
MOVE 'GET' TO WS-FUNCTION
MOVE MQCC-OK TO WS-CCODE-VALUE
MOVE MQRC-NONE TO WS-RCODE-VALUE
MOVE 500 TO WS-BUFFER-LENGTH
MOVE MQGMO-ACCEPT-TRUNCATED-MSG
    TO MQGMO-OPTIONS
MOVE LOW-VALUES TO MQMD-MSGID
    MQMD-CORRELID
*
CALL 'MQGET' USING WS-HCONN-ADDR-AREA
                  WS-HOBJ-ADDR-AREA
                  WS-MSG-DESCRIPTOR
                  WS-GET-OPTIONS
                  WS-BUFFER-L-AREA
                  WS-BUFFER-AREA
                  WS-DATA-L-AREA
                  WS-CCODE-ADDR-AREA
                  WS-RCODE-ADDR-AREA
*
IF (WS-CCODE-VALUE NOT EQUAL ZERO)
THEN
    IF WS-RCODE-VALUE EQUAL 2079
    THEN
        SET WS-TRUNCATED-MESSAGES TO TRUE
    ELSE
        IF WS-RCODE-VALUE EQUAL 2033
        THEN
            SET WS-END-OF-MESSAGES TO TRUE
            GO TO 3000-GET-EOF
        ELSE
            GO TO 9900-ERR-DISPLAY
    END-IF
END-IF
*
END-IF
*-- --CHECK IF "NEGATIVE " PROCESSING OPTION SPECIFIED
IF WS-DATA-SYNC-FLAG NOT EQUAL '-'
THEN
    EXEC CICS SYNCPOINT
    END-EXEC
END-IF
ADD +1 TO WS-COUNT
END-PERFORM.
*
-----*
3000-GET-EOF.
*--MQCLOSE QUEUE TO QM
MOVE 'CLOSE' TO WS-FUNCTION.
MOVE ZERO TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQCLOSE' USING WS-HCONN-ADDR-AREA
                    WS-HOBJ-ADDR-AREA
                    WS-Q-OPEN-OPTIONS
                    WS-CCODE-ADDR-AREA

```

```

WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.
*--MQDISC FROM QM
MOVE 'DISCONN' TO WS-FUNCTION.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQDISC' USING
                    WS-HCONN-ADDR-AREA
                    WS-CCODE-ADDR-AREA
                    WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.
*
-----*
3000-EXIT.
EXIT.
EJECT
*
-----*
4000-PUT1-MESSAGES.
*
-----*
* PURPOSE: CONNECT , OPEN
* PUT
* CLOSE, DISCONNECT
*
-----*
*--MQCONNECT TO QM
MOVE 'CONNECT' TO WS-FUNCTION.
MOVE SPACES TO WS-QM-NAME-CONNECT.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HCONN-VALUE TO NULL.
CALL 'MQCONN' USING WS-QM-NAME-AREA
                   WS-HCONN-ADDR-AREA
                   WS-CCODE-ADDR-AREA
                   WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.
*
-----*
PERFORM WS-PROCESS-TIMES TIMES
*
*--CHECK IF MUST PUT TIME STAMP ON MESSAGE
IF WS-PUT-TIMESTAMP
THEN
    PERFORM 8000-GET-TIME-STAMP
    MOVE WS-TIMESTAMP-VALUE TO WS-BUFFER-TS
    MOVE LENGTH OF WS-BUFFER-TS
        TO WS-BUFFER-LENGTH
    ADD WS-MSG-LENGTH TO WS-BUFFER-LENGTH
    MOVE WS-MSG-AREA TO WS-BUFFER-TEXT
    ELSE
        MOVE WS-MSG-LENGTH TO WS-BUFFER-LENGTH
        MOVE WS-MSG-AREA TO WS-BUFFER-AREA
    END-IF
*
*--MQPUT1 QUEUE TO QM
MOVE 'PUT1' TO WS-FUNCTION
MOVE MQOO-OUTPUT TO MQPMO-OPTIONS
MOVE SPACES TO MQOD-OBJECTQMGRNAME
MOVE WS-DATA-QUEUE TO MQOD-OBJECTNAME
MOVE MQCC-OK TO WS-CCODE-VALUE
MOVE MQRC-NONE TO WS-RCODE-VALUE
CALL 'MQPUT1' USING WS-HCONN-ADDR-AREA
                   WS-Q-NAME-AREA
                   WS-MSG-DESCRIPTOR
                   WS-PUT-OPTIONS
                   WS-BUFFER-L-AREA
                   WS-BUFFER-AREA
                   WS-CCODE-ADDR-AREA
                   WS-RCODE-ADDR-AREA
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY

```



```

END-IF
*
END-PERFORM.
*-----*
*--MQDISC FROM QM
MOVE 'DISCONN' TO WS-FUNCTION.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQDISC' USING
        WS-HCONN-ADDR-AREA
        WS-CCODE-ADDR-AREA
        WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
        GO TO 9900-ERR-DISPLAY.
*-----*
4000-EXIT.
EXIT.
EJECT
*-----*
5000-GETD-MESSAGES.
*-----*
* PURPOSE: CONNECT , OPEN
* GET
* CLOSE, DISCONNECT
*-----*
*--MQCONNECT TO QM
MOVE 'CONNECT' TO WS-FUNCTION.
MOVE SPACES TO WS-QM-NAME.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HCONN-VALUE TO NULL.
CALL 'MQCONN' USING WS-QM-NAME-AREA
        WS-HCONN-ADDR-AREA
        WS-CCODE-ADDR-AREA
        WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
        GO TO 9900-ERR-DISPLAY.
*
*--MQOPEN QUEUE TO QM
MOVE 'OPEN' TO WS-FUNCTION.
MOVE MQ00-BROWSE TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE SPACES TO MQ0D-OBJECTQMGRNAME.
MOVE WS-DATA-QUEUE TO MQ0D-OBJECTNAME.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HOBJ-VALUE TO NULL.
CALL 'MQOPEN' USING WS-HCONN-ADDR-AREA
        WS-Q-NAME-AREA
        WS-Q-OPEN-OPTIONS
        WS-HOBJ-ADDR-AREA
        WS-CCODE-ADDR-AREA
        WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
        GO TO 9900-ERR-DISPLAY.
*
*-----*
PERFORM WS-PROCESS-TIMES TIMES
*
*--MQGET TO QUEUE TO QM
MOVE 'GET' TO WS-FUNCTION
MOVE MQCC-OK TO WS-CCODE-VALUE
MOVE MQRC-NONE TO WS-RCODE-VALUE
MOVE 500 TO WS-BUFFER-LENGTH
MOVE MQGMO-BROWSE-FIRST TO MQGMO-OPTIONS
ADD MQGMO-ACCEPT-TRUNCATED-MSG
        TO MQGMO-OPTIONS
MOVE LOW-VALUES TO MQMD-MSGID
        MQMD-CORRELID
*
CALL 'MQGET' USING WS-HCONN-ADDR-AREA
        WS-HOBJ-ADDR-AREA
        WS-MSG-DESCRIPTOR
        WS-GET-OPTIONS
        WS-BUFFER-L-AREA
        WS-BUFFER-AREA
        WS-DATA-L-AREA
        WS-CCODE-ADDR-AREA
        WS-RCODE-ADDR-AREA
*
IF (WS-CCODE-VALUE NOT EQUAL ZERO)
THEN
        IF WS-RCODE-VALUE EQUAL 2079
        THEN
                SET WS-TRUNCATED-MESSAGES TO TRUE
        ELSE
                IF WS-RCODE-VALUE EQUAL 2033
                THEN
                        SET WS-END-OF-MESSAGES TO TRUE
                        GO TO 5000-GET-EOF
                ELSE
                        GO TO 9900-ERR-DISPLAY
        END-IF
END-IF
*
MOVE ZERO TO MQMD-REPORT
*--MQGET TO QUEUE TO QM W/ DELETE UNDER CURSOR
MOVE 'GET' TO WS-FUNCTION
MOVE MQCC-OK TO WS-CCODE-VALUE
MOVE MQRC-NONE TO WS-RCODE-VALUE
MOVE MQGMO-MSG-UNDER-CURSOR TO MQGMO-OPTIONS
MOVE 500 TO WS-BUFFER-LENGTH
MOVE LOW-VALUES TO MQMD-MSGID
        MQMD-CORRELID
*
CALL 'MQGET' USING WS-HCONN-ADDR-AREA
        WS-HOBJ-ADDR-AREA
        WS-MSG-DESCRIPTOR
        WS-GET-OPTIONS
        WS-BUFFER-L-AREA
        WS-BUFFER-AREA
        WS-DATA-L-AREA
        WS-CCODE-ADDR-AREA
        WS-RCODE-ADDR-AREA
*
IF (WS-CCODE-VALUE NOT EQUAL ZERO)
THEN
        IF WS-RCODE-VALUE EQUAL 2079
        THEN
                SET WS-TRUNCATED-MESSAGES TO TRUE
        ELSE
                GO TO 9900-ERR-DISPLAY
        END-IF
END-IF
*
*--ADDED 4/ 5/93
*-- --CHECK IF "NEGATIVE " PROCESSING OPTION SPECIFIED
IF WS-DATA-SYNC-FLAG NOT EQUAL '-'
THEN
        EXEC CICS SYNCPOINT
        END-EXEC
END-IF
ADD +1 TO WS-COUNT
*
END-PERFORM.
*-----*
5000-GET-EOF.
*
*--MQCLOSE QUEUE TO QM
MOVE 'CLOSE' TO WS-FUNCTION.
MOVE ZERO TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQCLOSE' USING WS-HCONN-ADDR-AREA
        WS-HOBJ-ADDR-AREA
        WS-Q-OPEN-OPTIONS
        WS-CCODE-ADDR-AREA
        WS-RCODE-ADDR-AREA.
*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
        GO TO 9900-ERR-DISPLAY.

```

```

*--MQDISC FROM QM
  MOVE 'DISCONN' TO WS-FUNCTION.
  MOVE MQCC-OK TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  CALL 'MQDISC' USING
      WS-HCONN-ADDR-AREA
      WS-CCODE-ADDR-AREA
      WS-RCODE-ADDR-AREA.
*
  IF WS-CCODE-VALUE NOT EQUAL ZERO
  THEN
    GO TO 9900-ERR-DISPLAY.
-----*
5000-EXIT.
  EXIT.
  EJECT
-----*
6000-PUT-WITH-REPLY.
-----*
* PURPOSE: CONNECT , OPEN
* PUT
* CLOSE, DISCONNECT
-----*
*
*--MQCONNECT TO QM
  MOVE 'CONNECT' TO WS-FUNCTION.
  MOVE SPACES TO WS-QM-NAME-CONNECT.
  MOVE MQCC-OK TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  SET WS-HCONN-VALUE TO NULL.
  CALL 'MQCONN' USING WS-QM-NAME-AREA
      WS-HCONN-ADDR-AREA
      WS-CCODE-ADDR-AREA
      WS-RCODE-ADDR-AREA.
*
  IF WS-CCODE-VALUE NOT EQUAL ZERO
  THEN
    GO TO 9900-ERR-DISPLAY.
*
*--MQOPEN QUEUE FOR REPLY QUEUE
  MOVE 'OPEN' TO WS-FUNCTION.
  MOVE MQ00-INPUT-SHARED TO WS-Q-OPEN-OPTIONS-VALUE.
  MOVE MQMT-REQUEST TO MQMD-MSGTYPE.
  MOVE SPACES TO MQMD-REPLYTOQMGR.
  MOVE SPACES TO MQMD-REPLYTOQ.

  MOVE SPACES TO MQOD-OBJECTQMGRNAME.
  MOVE WS-REPLY-Q TO MQOD-OBJECTNAME.
  MOVE MQCC-OK TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  SET WS-HOBJ-VALUE-REPLY TO NULL.
  CALL 'MQOPEN' USING WS-HCONN-ADDR-AREA
      WS-Q-NAME-AREA
      WS-Q-OPEN-OPTIONS
      WS-HOBJ-ADDR-AREA-REPLY
      WS-CCODE-ADDR-AREA
      WS-RCODE-ADDR-AREA.
*
  IF WS-CCODE-VALUE NOT EQUAL ZERO
  THEN
    GO TO 9900-ERR-DISPLAY.
-----*
  PERFORM WS-PROCESS-TIMES TIMES
*--CHECK IF MUST PUT TIME STAMP ON MESSAGE
  IF WS-PUT-TIMESTAMP
  THEN
    PERFORM 8000-GET-TIME-STAMP
    MOVE WS-TIMESTAMP-VALUE TO WS-BUFFER-TS
    ADD WS-MSG-LENGTH LENGTH OF WS-BUFFER-TS
      GIVING WS-BUFFER-LENGTH
    MOVE WS-MSG-AREA TO WS-BUFFER-TEXT
  ELSE
    MOVE WS-MSG-LENGTH TO WS-BUFFER-LENGTH
    MOVE WS-MSG-AREA TO WS-BUFFER-AREA
  END-IF
*
*--MQPUT1 QUEUE TO QM
      MOVE 'PUT1' TO WS-FUNCTION
      MOVE MQMT-REPLY TO MQMD-MSGTYPE
      MOVE SPACES TO MQMD-REPLYTOQMGR
      MOVE WS-REPLY-Q TO MQMD-REPLYTOQ

      MOVE MQ00-OUTPUT TO MQPMO-OPTIONS
      MOVE SPACES TO MQOD-OBJECTQMGRNAME
      MOVE WS-DATA-QUEUE TO MQOD-OBJECTNAME
      MOVE MQCC-OK TO WS-CCODE-VALUE
      MOVE MQRC-NONE TO WS-RCODE-VALUE
      CALL 'MQPUT1' USING WS-HCONN-ADDR-AREA
          WS-Q-NAME-AREA
          WS-MSG-DESCRIPTOR
          WS-PUT-OPTIONS
          WS-BUFFER-L-AREA
          WS-BUFFER-AREA
          WS-CCODE-ADDR-AREA
          WS-RCODE-ADDR-AREA
*
  IF WS-CCODE-VALUE NOT EQUAL ZERO
  THEN
    GO TO 9900-ERR-DISPLAY
  END-IF
*--SYNPOINT PUT SO ECHO CAN GET IT
  EXEC CICS SYNCPOINT
  END-EXEC
*--MQGET TO QUEUE TO QM
  MOVE 'GET' TO WS-FUNCTION
  MOVE MQMT-REQUEST TO MQMD-MSGTYPE
  MOVE LOW-VALUES TO MQMD-MSGID
      MQMD-CORRELID
  MOVE SPACES TO MQMD-REPLYTOQMGR

  MOVE SPACES TO MQMD-REPLYTOQ

  MOVE MQCC-OK TO WS-CCODE-VALUE
  MOVE MQRC-NONE TO WS-RCODE-VALUE
  MOVE 500 TO WS-BUFFER-LENGTH
  MOVE MQGMO-ACCEPT-TRUNCATED-MSG
      TO MQGMO-OPTIONS
  ADD MQGMO-WAIT
      TO MQGMO-OPTIONS

  MOVE LOW-VALUES TO MQMD-MSGID
      MQMD-CORRELID
*--WAIT 30 SECONDS (IE, 30,000 MILL-SECONDS)
  MOVE +30000 TO MQGMO-WAITINTERVAL
*
  CALL 'MQGET' USING WS-HCONN-ADDR-AREA
      WS-HOBJ-ADDR-AREA-REPLY
      WS-MSG-DESCRIPTOR
      WS-GET-OPTIONS
      WS-BUFFER-L-AREA
      WS-BUFFER-AREA
      WS-DATA-L-AREA
      WS-CCODE-ADDR-AREA
      WS-RCODE-ADDR-AREA
*
  IF (WS-CCODE-VALUE NOT EQUAL ZERO)
  THEN
    IF WS-RCODE-VALUE EQUAL 2079
    THEN
      SET WS-TRUNCATED-MESSAGES TO TRUE
    ELSE
      IF WS-RCODE-VALUE EQUAL 2033
      THEN
        SET WS-END-OF-MESSAGES TO TRUE
        GO TO 6000-PUT-WITH-EOF
      ELSE
        GO TO 9900-ERR-DISPLAY
      END-IF
    END-IF
  ADD +1 TO WS-COUNT
*--SYNPOINT PUT SO ECHO CAN GET IT
  EXEC CICS SYNCPOINT
  END-EXEC

```

```

END-PERFORM.
*-----*
6000-PUT-WITH-EOF.

*--MQCLOSE QUEUE TO QM
MOVE 'CLOSE' TO WS-FUNCTION.
MOVE ZERO TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQCLOSE' USING WS-HCONN-ADDR-AREA
                     WS-HOBJ-ADDR-AREA-REPLY
                     WS-Q-OPEN-OPTIONS
                     WS-CCODE-ADDR-AREA
                     WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.

*--MQDISC FROM QM
MOVE 'DISCONN' TO WS-FUNCTION.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQDISC' USING WS-HCONN-ADDR-AREA
                   WS-CCODE-ADDR-AREA
                   WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.

*-----*
6000-EXIT.
EXIT.
EJECT

*-----*
*-- INQ function added.
6500-INQ-MESSAGES.
*-----*
* PURPOSE: CONNECT , OPEN
* INQ
* CLOSE, DISCONNECT
*-----*
*--MQCONNECT TO QM
MOVE 'CONNECT' TO WS-FUNCTION.
MOVE SPACES TO WS-QM-NAME-CONNECT.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HCONN-VALUE TO NULL.
CALL 'MQCONN' USING WS-QM-NAME-AREA
                  WS-HCONN-ADDR-AREA
                  WS-CCODE-ADDR-AREA
                  WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.

*
*--MQOPEN QUEUE TO QM
MOVE 'OPEN' TO WS-FUNCTION.
MOVE MQ00-INQUIRE TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE SPACES TO MQ00-OBJECTQMGRNAME.
MOVE WS-DATA-QUEUE TO MQ00-OBJECTNAME.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HOBJ-VALUE TO NULL.
CALL 'MQOPEN' USING WS-HCONN-ADDR-AREA
                  WS-Q-NAME-AREA
                  WS-Q-OPEN-OPTIONS
                  WS-HOBJ-ADDR-AREA
                  WS-CCODE-ADDR-AREA
                  WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.

*--SETUP INQ PARMS

```

```

MOVE MQCA-Q-DESC TO MQI-SELECTOR-ENTRY (1).
MOVE MQCA-Q-NAME TO MQI-SELECTOR-ENTRY (2).
MOVE MQIA-INHIBIT-PUT TO MQI-SELECTOR-ENTRY (3).
MOVE MQIA-Q-TYPE TO MQI-SELECTOR-ENTRY (4).
MOVE MQIA-MAX-MSG-LENGTH TO MQI-SELECTOR-ENTRY (5).
MOVE +5 TO WS-SELECTOR-COUNT.
MOVE LENGTH OF WS-CHAR-ATTR TO
                     WS-CHAR-ATTR-LENGTH

*
*--MQPUT TO QUEUE TO QM
MOVE 'INQ' TO WS-FUNCTION.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQINQ' USING WS-HCONN-ADDR-AREA
                  WS-HOBJ-ADDR-AREA
                  WS-SELECTOR-COUNT
                  MQI-SELECTOR
                  MQI-IN-ATTR-COUNT
                  MQI-IN-ATTR
                  MQI-CHAR-ATTR-LENGTH
                  MQI-CHAR-ATTR
                  WS-CCODE-ADDR-AREA
                  WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY
ELSE
    STRING 'NAME: ' WS-A-Q-N DELIMITED BY SIZE
    INTO WS-OK-MSG-7
    END-STRING
    STRING 'DESC: ' WS-A-DESC DELIMITED BY SIZE
    INTO WS-OK-MSG-8
    END-STRING
    MOVE MQI-IN-ATTR-ENTRY(1) TO WS-I-INHIBIT
    MOVE MQI-IN-ATTR-ENTRY(2) TO WS-I-TYPE
    MOVE MQI-IN-ATTR-ENTRY(3) TO WS-I-MAXL
    MOVE WS-INQ-DETS TO WS-OK-MSG-9

END-IF

*
*--MQCLOSE QUEUE TO QM
MOVE 'CLOSE' TO WS-FUNCTION.
MOVE ZERO TO WS-Q-OPEN-OPTIONS-VALUE.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQCLOSE' USING WS-HCONN-ADDR-AREA
                   WS-HOBJ-ADDR-AREA
                   WS-Q-OPEN-OPTIONS
                   WS-CCODE-ADDR-AREA
                   WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.

*--MQDISC FROM QM
MOVE 'DISCONN' TO WS-FUNCTION.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
CALL 'MQDISC' USING
                     WS-HCONN-ADDR-AREA
                     WS-CCODE-ADDR-AREA
                     WS-RCODE-ADDR-AREA.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
THEN
    GO TO 9900-ERR-DISPLAY.

*-----*
6500-EXIT.
EXIT.
*-----*
7000-SEND-TOTALS.
*-----*
*--GET DURATION TIME
EXEC CICS ASKTIME
      ABSTIME(WS-ABSTIME2)
END-EXEC.

SUBTRACT WS-ABSTIME FROM WS-ABSTIME2.
EXEC CICS FORMATTIME

```

```

        ABSTIME (WS-ABSTIME2)
        TIME (WS-DURATION-SECS)
        TIMESEP(':')
    END-EXEC.
*
    MOVE WS-COUNT          TO WS-OK-MESSAGES.
    MOVE WS-DURATION-SECS TO WS-OK-TIME.
    MOVE WS-DATA-QUEUE     TO WS-OK-QUEUE.
    IF WS-PUT-WITH-REPLY
    THEN
        MOVE WS-REPLY-Q      TO WS-OK-QUEUE-REPLY
        MOVE WS-OK-STATS-LINE-2 TO WS-OK-MSG-2.
*-- --MOVE REST
    MOVE WS-OK-STATS-LINE-1 TO WS-OK-MSG-1.
    MOVE WS-OK-STATS-LINE-3 TO WS-OK-MSG-3.
    MOVE WS-OK-STATS-LINE-4 TO WS-OK-MSG-4.
*-- --CHECK IF ANY ERRORS
    IF WS-END-OF-MESSAGES
    THEN
        MOVE 'NO MORE MESSAGES' TO WS-OK-MSG-5.
*
    IF WS-TRUNCATED-MESSAGES
    THEN
        MOVE 'TRUNCATED MESSAGES' TO WS-OK-MSG-6.
*
    IF WS-ERR-MSG
    EXEC CICS SEND
        FROM (WS-ALL-MSG)
        LENGTH (LENGTH OF WS-ALL-MSG)
        ERASE
    END-EXEC
    ELSE
    EXEC CICS SEND
        FROM (WS-OK-MSG)
        LENGTH (LENGTH OF WS-OK-MSG)
        ERASE
    END-EXEC.
*-----*
    EJECT
*-----*
    8000-GET-TIME-STAMP.
*-----*
    EXEC CICS ASKTIME
        ABSTIME(WS-ABSTIME)
    END-EXEC.
*
    EXEC CICS FORMATTIME
        ABSTIME(WS-ABSTIME)
        YYMMDD (WS-DATE-YYMMDD)
    END-EXEC.
*
    MOVE EIBTIME TO WS-TIME-9.
*
    MOVE WS-DATE-YYMMDD TO WS-TIMESTAMP-DATE.
    MOVE WS-TIME-HHMMSS TO WS-TIMESTAMP-TIME.
*-----*
    EJECT
*-----*
    9900-ERR-DATA.
*-----*
*--ERROR IN "GET" DATA
    SET WS-ERR-MSG TO TRUE.
    MOVE WS-DATA-LENGTH-USER TO WS-ERR-DATA-LENGTH.
    MOVE WS-BUFFER-AREA TO WS-ERR-DATA-AREA.
    MOVE WS-ERR-DATA TO WS-ERR-LINES.
*--IF STARTED..SEND MESSAGE
    IF WS-STARTED
    THEN
        PERFORM 9999-ERROR-WRITE.
*

```

```

        GO TO 0000-SEND-TOTALS.
    EJECT
*-----*
    9900-ERR-DISPLAY.
*-----*
*--ERROR IN "MQ" VERB
*
    SET WS-ERR-MSG TO TRUE.
    MOVE WS-CCODE-VALUE TO WS-ERR-DISPLAY-CCODE.
    MOVE WS-RCODE-VALUE TO WS-ERR-DISPLAY-RCODE.
*
    MOVE WS-ERR-DISPLAY TO WS-ERR-LINES.
*
*--IF STARTED..SEND MESSAGE
    IF WS-STARTED
    THEN
        PERFORM 9999-ERROR-WRITE.
*
    GO TO 0000-SEND-TOTALS.
    EJECT
*-----*
* ERROR HANDLING
*-----*
* COPY MQIERRCD.
*/INCLUDE MQIERRCD
*-----*
* ERROR PROCESSING - CODE PROCESSING - MQIERRCD
*-----*
    9999-ERROR-WRITE.
    EXEC CICS WRITEQ TD
        QUEUE (ENV-II-ERROR-TD)
        FROM (ERR-HANDLER-COMMAREA)
        LENGTH (LENGTH OF ERR-HANDLER-COMMAREA) NOHANDLE
    END-EXEC.
*--IF ERROR IN ERROR TD .. PUT TO CSMT
*WKH IF EIBRCODE NOT EQUAL LOW-VALUES
*-----*
    EJECT
*-----*
    9999-CONVERT-ERROR-INFO.
*-----*
    MOVE EIBTRNID TO ERR-TRANID.
    MOVE EIBTRMID TO ERR-TERMID.
    MOVE EIBTASKN TO ERR-TASKNO.
    MOVE WS-ABSTIME TO ERR-ABSTIME.
*
    MOVE EIBFN TO ERR-DEBUG-EIBFN.
    MOVE EIBRCODE TO ERR-DEBUG-EIBRCODE.
    MOVE EIBSRCE TO ERR-DEBUG-EIBSRCE.
*
    MOVE EIBRESP TO ERR-DEBUG-EIBRESP.
    MOVE EIBRESP2 TO ERR-DEBUG-EIBRESP2.
    MOVE EIBERRCD TO ERR-DEBUG-EIBERRCD.
*-----*
    EJECT
*-----*
* ERROR PROCESSING - ABEND PROCESSING
*-----*
    9999-ABEND-CONDITION.
    MOVE ERR-CICS-ABEND TO ERR-CODE.
    PERFORM 9999-CONVERT-ERROR-INFO.
*--ASSIGN INFO
    EXEC CICS ASSIGN ABCODE (ERR-DEBUG-ABEND)
    END-EXEC.
*--USER CODE MUST FOLLOW THIS STATEMENT *****
    9999-ABEND-USER-CODE.
*
    GO TO 0000-RETURN.

```

Sample program TTPTST3.Z

This program is a test facility for putting and getting messages, by starting the sample transaction TST2 (program ID TTPTST2). It can be invoked:

- By passing data, using a CICS “START” command.
- By the terminal input “TST3”, which produces the screen requesting more input shown in Figure 44.

```

06/22/93      IBM MQSeries for VSE/ESA Version 1      IYZFZSI3
14:43:13      *** Test System Programs 3 - STARTS ***  ISC2
                                                    A803

                Async TASK Information

Number of tasks.....:
Message Processing Information
Number of messages..:
Function.....:                P=PUT, or G= GET
PUT queue name.....:
PUT message size....:
PUT message.....:
PUT TimeStamp.....:          Y=Yes, N=No
ENTER START VALUES.

ENTER = Process                                PF3=Quit
    
```

Figure 44. Test System Programs 3 - start

On this screen the fields are:

Number of tasks	The number of asynchronous tasks (TST2 transactions).
Number of messages	The number of messages to be sent and received.
Function	Specify “P” to put message, “G” to get message.
PUT queue name	The queue name to be PUT or GET.
PUT message size	For the PUT function only, to specify the size of the message. If the PUT timestamp option is selected, the message is 16 characters larger than the PUT message size.
PUT message	The content of the message.
PUT TimeStamp	For the PUT function only, to put a time stamp in the message in the format YYMMDDHHMMSS. If you specify this option, the actual message size is 16 characters larger than the size specified in the PUT message size.

```

IDENTIFICATION DIVISION.
PROGRAM-ID.    TTPTST3.
DATE-COMPILED.
*-----*
* SUMMARY CHANGES:                TRACE POINT CODE: 109
*-----*
* CHANGE  DATE      PGM  COMMENT
*-----*
* TEST PROGRAMS TO START ASYNC TASKS
*-----*
                                                    *
                                                    *
                                                    *
*-----*
* PURPOSE: START ASYNC TASK      DEFINITION
*-----*
* COPYBOOKS:  TTMTST3 - COBOL MAP SYMBOLIC
*              MQIMTP - MASTER TERMINAL COMMAREA
*              TTITST2 - TTPTST2 COMMAREA FOR STARTS
*              MQIERRWS - ERROR VALUES
*              MQIERRCD - ERROR CODE
*              TTETST3 - ERROR MESSAGES
*              MQIENV - ENVIRONMENT
*              DFHAID - 3270 AID DEFINITION
*              DFHBMSCA - 3270 BMS CONTROL CHARACTERS
*-----*
    
```

TTPTST3.Z

```

* TRANSACTION:  TST3 - MASTER TERMINAL (UPDATE )          *
*                                                       *
* MAPSET:       TTMTST3                                  *
* MAPS:        MAIN  - MAIN                              *
*                                                       *
* SUMMARY CHANGES:                                     *
*                                                       *
-----*
      EJECT
-----*
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
-----*
*   COPY COPYRWS.
-----*
* COPYRIGHT WORKING STORAGE FOR COBOL MODULES          *
-----*
01  FILLER.
    05 FILLER                PIC X(72) VALUE
      'Licensed Materials - Property of IBM'.
    05 FILLER                PIC X(72) VALUE SPACES.
    05 FILLER                PIC X(72) VALUE
      '5686-A06 '.
    05 FILLER                PIC X(72) VALUE SPACES.
    05 FILLER                PIC X(72) VALUE
      '(C) Copyright IBM Corp. 1998      All Rights Reserve
-      'd'.
    05 FILLER                PIC X(72) VALUE SPACES.
    05 FILLER                PIC X(72) VALUE
      'US Government Users Restricted Rights - Use, duplication or
-      ' '.
    05 FILLER                PIC X(72) VALUE
      'disclosure restricted by GSA ADP Schedule Contract with IBM
-      ' Corp.'.
-----*
*Debugging eyecatcher information for start of WORKING-STORAGE.
-----*
01  WS-RWS-PROGRAM-NAME1    PIC X(8).
01  FILLER                  PIC X(16) VALUE
      ' Version V2.1.0'.
01  WS-RWS-WHEN-COMPILED   PIC X(21).
01  FILLER                  PIC X(7) VALUE '====='.
-----*

01  WS-VALUES.
    05 WS-CONFIGURATION-ADDRESS USAGE IS POINTER VALUE NULL.
    05 WS-REC-SIZE          PIC S9(4) COMP VALUE ZERO.
    05 WS-SS-STARTS        PIC 9(4) VALUE ZERO.
    05 WS-NUM              PIC 9(8) VALUE ZERO.
    05 WS-NUM4             PIC 9(4) VALUE ZERO.

    05 WS-APPLID          PIC X(8) VALUE SPACES.
    05 WS-SYSID          PIC X(4) VALUE SPACES.
    05 WS-STARTCD        PIC XX VALUE SPACES.
    88 WS-STARTED        VALUE 'SD'.

    05 WS-ABSTIME          PIC S9(15) COMP-3.
    05 WS-DATE-CCYYMMDD.
      10 WS-DATE-CC        PIC 99 VALUE ZERO.
      10 WS-DATE-YYMMDD.
        12 WS-DATE-YY      PIC 99 VALUE ZERO.
        12 WS-DATE-MM      PIC 99 VALUE ZERO.
        12 WS-DATE-DD      PIC 99 VALUE ZERO.
        12 FILLER          PIC XX VALUE ZERO.

    05 WS-UNPACK-TIME-9    PIC 9(07) VALUE ZEROES.
    05 WS-UNPACK-TIME-X REDEFINES WS-UNPACK-TIME-9.
      10 FILLER            PIC X(01).
      10 WS-TIME-HHMMSS.
        12 WS-TIME-HH      PIC X(02).
        12 WS-TIME-MM      PIC X(02).
        12 WS-TIME-SS      PIC X(02).

    05 WS-FORMATTED-TIME.
      10 WS-FORMAT-TIME-HH PIC X(02) VALUE SPACES.
      10 FILLER            PIC X(01) VALUE ':'.
      10 WS-FORMAT-TIME-MM PIC X(02) VALUE SPACES.
      10 FILLER            PIC X(01) VALUE ':'.
      10 WS-FORMAT-TIME-SS PIC X(02) VALUE SPACES.

    05 WS-FORMATTED-DATE.
      10 WS-FORMAT-DATE-MM PIC X(02) VALUE SPACES.
      10 FILLER            PIC X(01) VALUE '/'.

05  10 WS-FORMAT-DATE-DD   PIC X(02) VALUE SPACES.
05  10 FILLER              PIC X(01) VALUE '/'.
05  10 WS-FORMAT-DATE-YY   PIC X(02) VALUE SPACES.
05  10 WS-FORMAT-DATE-YY   PIC X(02) VALUE SPACES.
05  05 WS-TRAN-ID         PIC X(4) VALUE SPACES.
05  05 WS-EDIT-ERR-FLAG   PIC X(1) VALUE 'N'.
05  88 WS-EDIT-ERR        VALUE 'Y'.

05  05 WS-RECORD-FLAG     PIC X VALUE SPACES.
05  88 WS-RECORD-FOUND   VALUE 'T'.
05  88 WS-RECORD-NOT-FOUND VALUE 'F'.

05  05 WS-ERROR-MESSAGE   PIC X(79) VALUE SPACES.
05  05 WS-ERR-COUNT      PIC S9(4) COMP VALUE ZEROS.
05  05 WS-ERR-MAX        PIC S9(4) COMP VALUE +20.
05  05 WS-ERR-MESSAGE   VALUE SPACES.
05  10 WS-ERR-MSG        PIC X(79) OCCURS 20 TIMES.

-----*
      EJECT
-----*
      COPY DFHAID.
      EJECT
-----*
      COPY DFHBMSCA.
      EJECT
-----*
* BMS MAP
-----*
      COPY TTMTST3.
-----*
* TST2 COMMAREA
-----*
01  WS-TST2-COMMAREA.
*   COPY TTITST2.
-----*
* - BEGIN -      *** COPYBOOK: TTITST2 ***      - BEGIN - *
-----*
* 3/ 4/93 REV:
-----*
* MQPINIT1 COMMAREA
-----*

05  TST2-PASSED-INFO.
    10 TST2-FUNCTION        PIC X(4) VALUE 'PUT'.
    88 TST2-FUNCT-PUT      VALUE 'PUT'.
    88 TST2-FUNCT-GET      VALUE 'GET'.

    10 TST2-PUT-NUM-MSG    PIC S9(4) COMP VALUE ZERO.
    10 TST2-PUT-QUEUE-NAME PIC X(48) VALUE SPACES.
    10 TST2-PUT-MSG-SIZE   PIC S9(4) COMP VALUE ZERO.
    10 TST2-PUT-MSG        PIC X(48) VALUE SPACES.
    10 TST2-PUT-MSG-TIMESTAMP PIC X VALUE SPACES.
    88 TST2-PUT-MSG-W-TIMESTAMP VALUE 'Y'.

-----*
* - END -      *** COPYBOOK: TTITST2 ***      - END - *
-----*
      EJECT
-----*
* ENVIRONMENT VALUES
-----*
01  FILLER.
*   COPY MQICENV.
-----*
* - BEGIN -      *** COPYBOOK: MQICENV ***      - BEGIN - *
-----*
* ENVIRONMENT VALUE      - SYSTEM (ENV)
-----*

02  ENV-DEFINITION.
03  ENV-DATA-FOR-SYSTEM.
    05 ENV-PRODUCT-INSTALLED PIC X(4) VALUE 'MQI '.
    88 ENV-PRODUCT-MQM      VALUE 'MQM '.

    05 ENV-PRODUCT-RUNTIME  PIC X(4).
    88 ENV-PRODUCT-RT-MQM  VALUE 'MQM '.

05  ENV-LANG-INFO.
    10 ENV-LANGUAGE-FILE-CODE PIC 99 VALUE 01.
    10 ENV-LANGUAGE          PIC X(24)

```

```

VALUE 'ENGLISH'.
05 ENV-DATE-FORMAT PIC 99 VALUE 01.
88 ENV-DATE-MDDYY VALUE 01.
88 ENV-DATE-YMDD VALUE 02.
88 ENV-DATE-YDDMM VALUE 03.
88 ENV-DATE-YDDD VALUE 04.
88 ENV-DATE-DDMMY VALUE 05.

03 ENV-DATA-FOR-TRAN.
05 ENV-MASTER-TERMINAL-TRAN.
10 ENV-MT-MASTER-TASK-ID PIC X(4) VALUE 'MQMT'.
10 ENV-MT-CONFIG-TASK-ID PIC X(4) VALUE 'MQMC'.
10 ENV-MT-MONITOR-TASK-ID PIC X(4) VALUE 'MQMM'.
10 ENV-MT-OPER-TASK-ID PIC X(4) VALUE 'MQMO'.
10 ENV-MT-DISP-TASK-ID PIC X(4) VALUE 'MQBQ'.
10 ENV-MT-QUEUE-TASK-ID PIC X(4) VALUE 'MQMQ'.
10 ENV-MT-QUEUEI-TASK-ID PIC X(4) VALUE 'MQDQ'.
10 ENV-MT-COM-TASK-ID PIC X(4) VALUE 'MQMH'.
10 ENV-MT-COMI-TASK-ID PIC X(4) VALUE 'MQDH'.
10 ENV-MT-SYS-TASK-ID PIC X(4) VALUE 'MQMS'.
10 ENV-MT-SYSI-TASK-ID PIC X(4) VALUE 'MQDS'.
10 ENV-MT-MONQ-TASK-ID PIC X(4) VALUE 'MQQM'.
10 ENV-MT-MONC-TASK-ID PIC X(4) VALUE 'MQCM'.
10 ENV-MT-SS-TASK-ID PIC X(4) VALUE 'MQMA'.
10 ENV-MT-SC-TASK-ID PIC X(4) VALUE 'MQMB'.
10 ENV-MT-SI-TASK-ID PIC X(4) VALUE 'MQMI'.
10 ENV-MT-SR-TASK-ID PIC X(4) VALUE 'MQMR'.
10 ENV-MT-SD-TASK-ID PIC X(4) VALUE 'MQMD'.
10 FILLER PIC X(4) VALUE SPACES.
10 FILLER PIC X(4) VALUE SPACES.
10 FILLER PIC X(4) VALUE SPACES.

05 ENV-INTERNAL-ITEMS-TRAN.
10 ENV-II-MONITOR PIC X(4) VALUE 'MQSM'.
10 ENV-II-M-RECOVERY PIC X(4) VALUE 'MQSR'.
10 ENV-II-Q-RECOVERY PIC X(4) VALUE 'MQSQ'.
10 ENV-II-START-STOP PIC X(4) VALUE 'MQSS'.
10 ENV-II-TRAN-AIP2 PIC X(4) VALUE 'MQ02'.
10 ENV-II-TRAN-COM-CHECKP PIC X(4) VALUE 'MQCP'.
10 ENV-II-TRAN-QUE-DELETE PIC X(4) VALUE 'MQDD'.
10 ENV-II-TRAN-QUE-DEL-ALL PIC X(4) VALUE 'MQQA'.
10 ENV-II-TRAN-REORG-QUE PIC X(4) VALUE 'MQRG'.
10 ENV-II-TRAN-TCPIP PIC X(4) VALUE 'MQTL'.
10 FILLER PIC X(4) VALUE SPACES.

03 ENV-DATA-FOR-PROGRAMS.
05 ENV-MASTER-TERMINAL-PROGRAMS.
10 ENV-MT-MASTER-PROGRAM PIC X(8) VALUE 'MQPMT'.
10 ENV-MT-CONFIG-PROGRAM PIC X(8) VALUE 'MQPMC'.
10 ENV-MT-MONITOR-PROGRAM PIC X(8) VALUE 'MQPMON'.
10 ENV-MT-OPER-PROGRAM PIC X(8) VALUE 'MQPMOPR'.
10 ENV-MT-DISP-PROGRAM PIC X(8) VALUE 'MQPDISP'.
10 ENV-MT-QUEUE-PROGRAM PIC X(8) VALUE 'MQPQUE'.
10 ENV-MT-QUEUEI-PROGRAM PIC X(8) VALUE 'MQPQUEI'.
10 ENV-MT-COM-PROGRAM PIC X(8) VALUE 'MQPMCOM'.
10 ENV-MT-COMI-PROGRAM PIC X(8) VALUE 'MQPMCOMI'.
10 ENV-MT-SYS-PROGRAM PIC X(8) VALUE 'MQPMSYS'.
10 ENV-MT-SYSI-PROGRAM PIC X(8) VALUE 'MQPMSYSI'.
10 ENV-MT-MONQ-PROGRAM PIC X(8) VALUE 'MQPMMQ'.
10 ENV-MT-MONC-PROGRAM PIC X(8) VALUE 'MQPMMOC'.
10 ENV-MT-SS-PROGRAM PIC X(8) VALUE 'MQPMSS'.
10 ENV-MT-SC-PROGRAM PIC X(8) VALUE 'MQPMSC'.
10 ENV-MT-SI-PROGRAM PIC X(8) VALUE 'MQPMSI'.
10 ENV-MT-SR-PROGRAM PIC X(8) VALUE 'MQPMSNI'.
10 ENV-MT-SD-PROGRAM PIC X(8) VALUE 'MQPMD'.
10 ENV-MT-CMD-PROGRAM PIC X(8) VALUE 'MQPCMD'.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.

05 ENV-INTERNAL-ITEMS-PROGRAMS.
10 ENV-II-LINK-ERROR PIC X(8) VALUE 'MQPERR'.
10 ENV-II-LINK-EIB1 PIC X(8) VALUE 'MQPEIB1'.
10 ENV-II-LINK-AIP0 PIC X(8) VALUE 'MQPAIP0'.
10 ENV-II-LINK-AIP1 PIC X(8) VALUE 'MQPAIP1'.
10 ENV-II-LINK-AIP2 PIC X(8) VALUE 'MQPAIP2'.

10 ENV-II-LINK-ECHO PIC X(8) VALUE 'MQPECHO'.
10 ENV-II-LINK-FINDQ PIC X(8) VALUE 'MQPFINDQ'.
10 ENV-II-LINK-QUE1 PIC X(8) VALUE 'MQPQUE1'.

10 ENV-II-LINK-QUE2 PIC X(8) VALUE 'MQPQUE2'.
10 ENV-II-LINK-INIT1 PIC X(8) VALUE 'MQPINIT1'.
10 ENV-II-LINK-INIT2 PIC X(8) VALUE 'MQPINIT2'.
10 ENV-II-LINK-SSQ PIC X(8) VALUE 'MQPSSQ'.
10 ENV-II-LINK-SCHK PIC X(8) VALUE 'MQPSCHK'.
10 ENV-II-LINK-SREC PIC X(8) VALUE 'MQPSREC'.
10 ENV-II-LINK-ORECOVERY PIC X(8) VALUE 'MQPQREC'.
10 ENV-II-LINK-SENDER PIC X(8) VALUE 'MQPSEND'.
10 ENV-II-LINK-RECIEVER PIC X(8) VALUE 'MQPRECV'.
10 ENV-II-LINK-COM-CHECKP PIC X(8) VALUE 'MQPCCKPT'.
10 ENV-II-LINK-QUE-DELETE PIC X(8) VALUE 'MQPQDEL'.
10 ENV-II-LINK-SET-MAP PIC X(8) VALUE 'MQPSMAP'.
10 ENV-II-LINK-MQPVSAM PIC X(8) VALUE 'MQPVSAM'.

--- This module is no longer used. As a dirty fix the area has
--- been reused to hold the code page information.
* 10 ENV-II-LINK-LU21 PIC X(8) VALUE 'MQPLU21'.
10 ENV-CODE-PAGE PIC S9(4) COMP.
10 ENV-TCPIP-PORT-NUMBER PIC 9(8) COMP.
10 FILLER PIC XX.

10 ENV-II-LINK-REPORT PIC X(8) VALUE 'MQPRPRT'.
10 ENV-II-LINK-MQPCMD PIC X(8) VALUE 'MQPCMD'.
10 ENV-II-LINK-MQCLNT PIC X(8) VALUE 'MQPEXT1'.

03 ENV-DATA-FOR-MAPS.
05 ENV-MASTER-TERMINAL-MAPS.
10 ENV-MT-MASTER-MAPSCREEN PIC X(8) VALUE 'MQMMP'.
10 ENV-MT-CONFIG-MAPSCREEN PIC X(8) VALUE 'MQMCFG'.
10 ENV-MT-MONITOR-MAPSCREEN PIC X(8) VALUE 'MQMMON'.
10 ENV-MT-OPER-MAPSCREEN PIC X(8) VALUE 'MQMOPR'.
10 ENV-MT-DISP-MAPSCREEN PIC X(8) VALUE 'MQMDISP'.
10 ENV-MT-QUEUE-MAPSCREEN PIC X(8) VALUE 'MQMQUE'.
10 ENV-MT-QUEUEI-MAPSCREEN PIC X(8) VALUE 'MQMQUEI'.
10 ENV-MT-COM-MAPSCREEN PIC X(8) VALUE 'MQMCOM'.
10 ENV-MT-COMI-MAPSCREEN PIC X(8) VALUE 'MQMCOMI'.
10 ENV-MT-SYS-MAPSCREEN PIC X(8) VALUE 'MQMMSYS'.
10 ENV-MT-SYSI-MAPSCREEN PIC X(8) VALUE 'MQMMSYSI'.
10 ENV-MT-MONQ-MAPSCREEN PIC X(8) VALUE 'MQMMQ'.
10 ENV-MT-MONC-MAPSCREEN PIC X(8) VALUE 'MQMMOC'.
10 ENV-MT-SS-MAPSCREEN PIC X(8) VALUE 'MQMSS'.
10 ENV-MT-SC-MAPSCREEN PIC X(8) VALUE 'MQMSC'.
10 ENV-MT-SI-MAPSCREEN PIC X(8) VALUE 'MQMSI'.
10 ENV-MT-SR-MAPSCREEN PIC X(8) VALUE 'MQMSNI'.
10 ENV-MT-SD-MAPSCREEN PIC X(8) VALUE 'MQMDEL'.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.

03 ENV-DATA-FOR-CONSTANTS.
05 ENV-CONFIG-DDNAME PIC X(8) VALUE 'MQFCNFG'.
05 ENV-SYSTEM-NUMBER PIC 9(4) VALUE 1.
05 ENV-MASTER-TERMINAL-CONS.
10 ENV-MT-TITLE PIC X(40) VALUE
' IBM MQSeries for VSE/ESA Version 2 '.
05 ENV-INTERNAL-ITEMS-CONS.
10 ENV-II-ERROR-TD PIC X(4) VALUE 'MQER'.
10 ENV-II-ERROR-CSMT PIC X(4) VALUE 'CSMT'.
10 ENV-II-SYSTEM-ANCHOR PIC X(8) VALUE 'MQTAQM'.
10 ENV-II-SYSTEM-PREFIX PIC X(4) VALUE 'MQI'.
10 ENV-II-DUMPCODE PIC X(4) VALUE 'MQ??'.
10 ENV-II-ENQ-INIT1 PIC X(8) VALUE 'MQPINIT1'.
10 ENV-II-SYSTEM-ENVR PIC X(8) VALUE 'MQTENV'.
10 ENV-IT-UN-INIT-MSG PIC X(80) VALUE
'MQI900000: MQSERIES VSE ENVIRONMENT NOT INITIALIZED.'.
10 FILLER PIC X(80) VALUE SPACES.

-----*
* - END - *** COPYBOOK: MQICENV *** - END - *
-----*
EJECT
-----*
* FINDQ COMMAREA
-----*
01 WS-FINDQ.
* COPY MQIFINDQ.
-----*
* - BEGIN - *** COPYBOOK: MQIFINDQ *** - BEGIN - *
-----*

```

```

* 9/ 1/93 REV: *
*-----*
* FIND QUEUE CALL PARAMETERS. *
*-----*
02 FINDQ-CALL-PARAMETERS.

*--PASSED INFO...
03 FINDQ-PASSED-PARAMETERS.
05 FINDQ-CALL-TYPE PIC X VALUE SPACES.
88 FINDQ-QUEUE-LOOKUP VALUE 'Q'.
88 FINDQ-SYSTEM-STATUS-ONLY VALUE 'S'.

05 FILLER PIC X VALUE SPACES.
05 FINDQ-CALL-SYSTEM-NUM PIC 99 VALUE ZERO.

*-- --QUEUE INFO
05 FINDQ-QM-QUEUE-NAME.
10 FINDQ-QM-NAME PIC X(48) VALUE SPACES.
10 FINDQ-QUEUE-NAME PIC X(48) VALUE SPACES.

*--RETURN INFO
*-- --SYSTEM RETURN (ALWAYS RETURNED)
03 FINDQ-RETURNED-PARAMETERS.
05 FINDQ-SYSTEM-CODE PIC X VALUE SPACES.
88 FINDQ-SYSTEM-ACTIVE VALUE 'A'.
88 FINDQ-SYSTEM-INACTIVE VALUE 'I'.
88 FINDQ-SYSTEM-UN-INIT VALUE SPACE.

05 FILLER PIC XXX VALUE SPACES.
*-- -- --SYSTEM INFO (NOT SET IF SYSTEM UN-INIT)
05 FINDQ-DEFAULT-QM-INFO.
10 FINDQ-DEFAULT-NAME PIC X(48).
10 FINDQ-QM-DESCRIPTION PIC X(64).
10 FINDQ-DEFAULT-MAX-MSG PIC S9(8) COMP.
10 FINDQ-DEFAULT-MAX-CONN PIC S9(8) COMP.
10 FINDQ-DEFAULT-MAX-HANDLES PIC S9(8) COMP.
10 FINDQ-DEFAULT-MAX-WAIT-MON PIC S9(8) COMP.
10 FINDQ-DEFAULT-MAX-WAIT-REC PIC S9(8) COMP.
10 FINDQ-DEFAULT-MAX-REC-TASKS PIC S9(4) COMP.
10 FILLER PIC XX.
10 FINDQ-CONFIG-FILE PIC X(8).
88 FINDQ-CONFIG-FILE-OK VALUE 'MQFCNFG'.

10 FINDQ-DEADLETTER-NAME PIC X(48).
10 FINDQ-LOG-NAME PIC X(48).
10 FINDQ-AUDIT-NAME PIC X(48).
10 FINDQ-MONITOR-NAME PIC X(48).
10 FINDQ-ERROR-NAME PIC X(48).
10 FINDQ-MONITOR-SYS-FLAG PIC X.
88 FINDQ-MONITOR-ON VALUE 'Y'.

10 FINDQ-ERROR-TO-CSMT-FLAG PIC X.
88 FINDQ-ERROR-TO-CSMT VALUE 'Y', 'B'.
88 FINDQ-ERROR-TO-BOTH VALUE 'B'.

10 FILLER PIC XX.

*-- --QUEUE RETURN (ONLY RETURNED IF QUEUE REQUESTED)
05 FINDQ-QUEUE-CODE PIC X VALUE SPACES.
88 FINDQ-QUEUE-OK VALUE 'Y'.
88 FINDQ-QUEUE-NOT-FOUND VALUE SPACES.

05 FILLER PIC XXX VALUE SPACES.

*-- -- --ACTUAL MQI RETURN CODE
05 FINDQ-QUEUE-ERROR-CODE PIC S9(8) COMP VALUE ZERO.

*-- -- --QUEUE INFO (NOT RETURNED IF QUEUE NOT-FOUND)
05 FINDQ-RESOLVED-QM-QUEUE-NAME.
10 FINDQ-R-QM-NAME PIC X(48) VALUE SPACES.
10 FINDQ-R-QUEUE-NAME PIC X(48) VALUE SPACES.

05 FINDQ-RESOLVED-LOCAL-NAME PIC X(48) VALUE SPACES.
05 FINDQ-QUEUE-DRQ-ITEM PIC S9(4) COMP VALUE ZERO.
05 FILLER PIC XX VALUE SPACES.

*-- -- -- --STATUS FROM DRQ
05 FINDQ-RESOLVE-STATUS.
10 FINDQ-R-INBOUND-STAT PIC XX VALUE SPACES.
10 FINDQ-R-OUTBOUND-STAT PIC XX VALUE SPACES.

*-- -- -- --ORIGINAL QUEUE VALUES
05 FINDQ-QUEUE-DATA.
10 FINDQ-ADDED-DATA.
15 FINDQ-ADDED-TIME PIC X(6).
15 FILLER PIC XX.
15 FINDQ-ADDED-DATE PIC X(8).
15 FINDQ-ADDED-TERMINID PIC X(8).
15 FINDQ-ADDED-USERID PIC X(3).
15 FILLER PIC X.

10 FINDQ-DESCRIPTION PIC X(64).
10 FINDQ-TYPE PIC X.
88 FINDQ-QUEUE-DEFINITION VALUE
'L', 'X', 'A', 'R', 'M'.
88 FINDQ-LOCAL-Q-ENTRY VALUE 'L'.
88 FINDQ-LOCAL-AIX-Q VALUE 'X'.
88 FINDQ-ALIAS-Q-ENTRY VALUE 'A'.
88 FINDQ-REMOTE-Q-ENTRY VALUE 'R'.
88 FINDQ-MODEL-Q-ENTRY VALUE 'M'.

10 FINDQ-TYPE-ALIAS PIC X.
88 FINDQ-ALIAS-QUEUE VALUE 'Q'.
88 FINDQ-ALIAS-MANAGER VALUE 'M'.
88 FINDQ-ALIAS-REPLY VALUE 'R'.

10 FILLER PIC XX.

10 FINDQ-ATTR-FLAGS.
15 FINDQ-INHIBIT-PUT-FLAG PIC X.
88 FINDQ-INHIBIT-PUT VALUE 'Y'.
15 FINDQ-INHIBIT-GET-FLAG PIC X.
88 FINDQ-INHIBIT-GET VALUE 'Y'.

15 FINDQ-PERSIST-FLAG PIC X.
88 FINDQ-PERSIST-DEFAULT VALUE 'Y'.

10 FILLER PIC X.

05 FINDQ-LOCAL-INFO.
10 FINDQ-DEFINITION-FLAG PIC X.
88 FINDQ-DEF-PERM VALUE 'Y'.
88 FINDQ-DEF-NOT-PERM VALUE 'N'.

10 FINDQ-USAGE-MODE-FLAG PIC X.
88 FINDQ-U-MODE-NORMAL VALUE 'N'.
88 FINDQ-U-MODE-TRANSM VALUE 'Y'.

10 FINDQ-SHAREABLE-FLAG PIC X.
88 FINDQ-SHARE-QUEUE VALUE 'Y'.
88 FINDQ-NON-SHARE-QUEUE VALUE 'N'.

10 FINDQ-TRIGGER-TYPE PIC X.
88 FINDQ-NO-TRIGGER VALUE SPACE.
88 FINDQ-TRIGGER-ON VALUE 'Y'.

*-----*
* - END - *** COPYBOOK: MQIFINDQ *** - END - *
*-----*

EJECT
*-----*
* COMMAREA
*-----*
* COPY MQIMTP.
*-----*
* COPYBOOK: MQIMTP
*
* FUNCTION: COMMAREA FOR MASTER TERMINAL TASK
*
*-----*
01 MTP-COMMAREA.
05 MTP-HEADER-FLAG PIC X(4) VALUE 'MQI '.
88 MTP-HEADER-OK VALUE 'MQI '.

05 MTP-MAIN-TASK PIC X(4) VALUE SPACES.
88 MTP-NO-RETURN-TASK VALUE SPACES.

05 MTP-ACTIVE-TASK PIC X(4) VALUE SPACES.

05 MTP-MAP-VALUE PIC X(8) VALUE 'MAIN'.
88 MTP-MAP-MAIN VALUE 'MAIN'.

```



```

      88 MTP-MAP-OPTIONS      VALUE 'OPTIONS'.
      88 MTP-MAP-QUEUE        VALUE 'QUEUE ' .
      88 MTP-MAP-LOCAL        VALUE 'LOCAL ' .
      88 MTP-MAP-QLIST        VALUE 'QLIST ' .
      88 MTP-MAP-REORG        VALUE 'REORG ' .

05 MTP-SCREEN-IND          PIC X      VALUE SPACE.
      88 MTP-SCREEN-FIRST     VALUE 'F'.
      88 MTP-SCREEN-RETURN    VALUE SPACE.
      88 MTP-SCREEN-SEND      VALUE 'S'.
      88 MTP-SCREEN-RECEIVE   VALUE 'R'.

05 MTP-MAP-FUNCTION       PIC X(8) VALUE 'DISPLAY'.
      88 MTP-MAP-DISPLAY      VALUE 'DISPLAY'.
      88 MTP-MAP-LIST         VALUE 'LIST'.
      88 MTP-MAP-ADD          VALUE 'ADD ' .
      88 MTP-MAP-UPDATE       VALUE 'UPDATE ' .
      88 MTP-MAP-DELETE       VALUE 'DELETE ' .

05 MTP-CONFIG-FILE       PIC X(8) VALUE SPACE.
05 MTP-SYSTEM-REC-FLAG   PIC X      VALUE SPACE.
      88 MTP-SYSTEM-REC-FOUND VALUE 'Y'.
      88 MTP-SYSTEM-REC-NOTFOUND VALUE 'N'.

05 MTP-CONFIRM-IND       PIC X      VALUE SPACE.
      88 MTP-CONFIRM          VALUE 'Y'.
      88 MTP-NO-CONFIRM       VALUE 'N'.

05 FILLER                PIC XX      VALUE SPACE.

*--CONFIGURATION DATA
05 MTP-CONFIG-DATA       PIC X(4) VALUE SPACES.

*--GENERAT EXTENDED DATA
05 MTP-EXTENDED-COMMAREA.
10 FILLER                PIC X(2000) VALUE SPACES.

*-----*
*-----*
EJECT
*-----*
* CONFIGURATION FILE
*-----*
* COPY MQICONFG.
*-----*
* - BEGIN -      *** COPYBOOK: MQICONFG ***      - BEGIN - *
*-----*
* CONFIGURATION FILE
*-----*
*-----*
* MAIN CONFIGURATION RECORD
*-----*
01 CONFIGURATION-RECORD VALUE SPACES.
03 FILLER                PIC X(2048).

*-----*
* ENVIRONMENT VALUE      - SYSTEM (ENV)
*-----*

01 ENVIRONMENT-RECORD
REDEFINES CONFIGURATION-RECORD.
03 ENV-RECORD-KEY.
05 ENV-RECORD-ID          PIC X(4).
08 RECORD-TYPE-IS-ENV    VALUE 'ENV'.
05 ENV-RECORD-VERSION    PIC 9(4).
05 ENV-RECORD-TYPE       PIC X(4).
08 ENV-TYPE-SYSTEM       VALUE 'SYS'.
08 ENV-TYPE-TRANSACTION   VALUE 'TRAN'.
08 ENV-TYPE-PROGRAM       VALUE 'PROG'.
08 ENV-TYPE-MAPS          VALUE 'MAPS'.
08 ENV-TYPE-CONSTANTS     VALUE 'CONS'.
05 ENV-FILLER            PIC X(88).

03 ENV-LAST-MAINTAINED-DATA.
05 ENV-LAST-TIME          PIC 9(6).
05 FILLER                 PIC XX.
05 ENV-LAST-DATE          PIC X(8).
05 ENV-LAST-TERMID        PIC X(4).
05 ENV-LAST-USERID        PIC X(8).

03 ENV-ADDED-MAINTAINED-DATA.
05 ENV-ADDED-TIME          PIC 9(6).
05 FILLER                 PIC XX.
05 ENV-ADDED-DATE          PIC X(8).
05 ENV-ADDED-TERMID        PIC X(4).
05 ENV-ADDED-USERID        PIC X(8).

03 ENV-DATA-AREA.
05 FILLER                 PIC X(1892).

*-----*
* SYSTEM DESCRIPTOR RECORD (SYS)
*-----*

01 SYSTEM-DESCRIPTOR-RECORD
REDEFINES CONFIGURATION-RECORD.
03 SYS-RECORD-KEY.
05 SYS-RECORD-ID          PIC X(4).
08 RECORD-TYPE-IS-SYS    VALUE 'SYS'.
05 SYS-RECORD-SYSTEM-NUMBER PIC 9(4).
05 SYS-RECORD-TYPE        PIC X(4).
08 SYS-TYPE-SYS          VALUE 'SYS'.
08 SYS-TYPE-QUE-MAX      VALUE 'QUEM'.
08 SYS-TYPE-QUE-DEFAULT  VALUE 'QUED'.
08 SYS-TYPE-COM-MAX      VALUE 'COMM'.
08 SYS-TYPE-COM-DEFAULT  VALUE 'COMD'.
08 SYS-TYPE-COM-PARM     VALUE 'COMP'.
05 SYS-FILLER            PIC X(88).

03 SYS-LAST-MAINTAINED-DATA.
05 SYS-LAST-TIME          PIC 9(6).
05 FILLER                 PIC XX.
05 SYS-LAST-DATE          PIC X(8).
05 SYS-LAST-TERMID        PIC X(4).
05 SYS-LAST-USERID        PIC X(8).

03 SYS-ADDED-MAINTAINED-DATA.
05 SYS-ADDED-TIME          PIC 9(6).
05 FILLER                 PIC XX.
05 SYS-ADDED-DATE          PIC X(8).
05 SYS-ADDED-TERMID        PIC X(4).
05 SYS-ADDED-USERID        PIC X(8).

03 SYS-DATA.
05 FILLER                 PIC X(1892).

*-----*
* QUEUE DESCRIPTOR RECORD (QDR)
*-----*

01 QUEUE-DESCRIPTOR-RECORD
REDEFINES CONFIGURATION-RECORD.
03 QDR-RECORD-KEY.
05 QDR-RECORD-ID          PIC X(4).
08 RECORD-TYPE-IS-QDR    VALUE 'QDR'.
05 QDR-RECORD-SYSTEM-NUMBER PIC 9(4).
05 QDR-OBJ-NAME           PIC X(48).
05 FILLER                 PIC X(44).

03 QDR-LAST-MAINTAINED-DATA.
05 QDR-LAST-TIME          PIC 9(6).
05 FILLER                 PIC XX.
05 QDR-LAST-DATE          PIC X(8).
05 QDR-LAST-TERMID        PIC X(4).
05 QDR-LAST-USERID        PIC X(8).

03 QDR-ADDED-MAINTAINED-DATA.
05 QDR-ADDED-TIME          PIC 9(6).
05 FILLER                 PIC XX.
05 QDR-ADDED-DATE          PIC X(8).
05 QDR-ADDED-TERMID        PIC X(4).
05 QDR-ADDED-USERID        PIC X(8).

03 QDR-DATA.
05 FILLER                 PIC X(1892).

*-----*
* COMMUNICATION (COM)
*-----*

```

```

01 COMMUNICATION-RECORD
  REDEFINES CONFIGURATION-RECORD.
03 COM-RECORD-KEY.
  05 COM-RECORD-ID          PIC X(4).
  88 RECORD-TYPE-IS-COM    VALUE 'COM'.
  05 COM-RECORD-SYSTEM-NUMBER PIC 9(4).
  05 COM-NAME                PIC X(20).
  05 COM-KEY-TYPE           PIC X.
  05 FILLER                  PIC X(71).

03 COM-LAST-MAINTAINED-DATA.
  05 COM-LAST-TIME         PIC 9(6).
  05 FILLER                 PIC XX.
  05 COM-LAST-DATE        PIC X(8).
  05 COM-LAST-TERMINID    PIC X(4).
  05 COM-LAST-USERID      PIC X(8).

03 COM-ADDED-MAINTAINED-DATA.
  05 COM-ADDED-TIME       PIC 9(6).
  05 FILLER                PIC XX.
  05 COM-ADDED-DATE       PIC X(8).
  05 COM-ADDED-TERMINID  PIC X(4).
  05 COM-ADDED-USERID    PIC X(8).

03 COM-DATA.
  05 FILLER                PIC X(1892).

*-----*
* TEXTUAL RECORD (TEXT)
*-----*

01 TEXT-RECORD
  REDEFINES CONFIGURATION-RECORD.
03 TEXT-RECORD-KEY.
  05 TEXT-RECORD-ID          PIC X(4).
  88 RECORD-TYPE-IS-TEXT    VALUE 'TEXT'.

  05 TEXT-RECORD-SYSTEM-NUMBER PIC 9(4).
  05 TEXT-RECORD-TYPE        PIC X(4).
  88 TEXT-TYPE-MESSAGES     VALUE 'MSGs'.
  88 TEXT-TYPE-MAPS         VALUE 'MAPS'.
  88 TEXT-TYPE-HELP         VALUE 'HELP'.

  05 TEXT-HELP-KEY.
  10 TEXT-HELP-SCREEN       PIC X(8).
  10 TEXT-HELP-FUNCTION     PIC X(40).
  10 TEXT-HELP-FUNCT-SCREEN-NUM PIC S9(4) COMP.

  05 TEXT-MAPS-KEY
  REDEFINES TEXT-HELP-KEY.
  10 TEXT-MAPS-SCREEN       PIC X(8).
  88 TEXT-MAPS-MAIN-TITLE   VALUE 'ALL'.

  10 TEXT-MAPS-MAPSET       PIC X(8).
  10 TEXT-MAPS-MAPSET-TYPE  PIC X(4).
  88 TEXT-MAPS-MAPSET-HEADER VALUE 'HEAD'.
  88 TEXT-MAPS-MAPSET-DATA  VALUE 'DATA'.
  88 TEXT-MAPS-MAPSET-MSGS  VALUE 'MSGs'.

  10 TEXT-MAPS-SCREEN-DATA-NUM PIC S9(4) COMP.

  05 TEXT-MESSAGE-KEY
  REDEFINES TEXT-HELP-KEY.
  10 TEXT-MESS-NUMBER       PIC 9(6).
  10 TEXT-MESS-RECORD-NUM   PIC S9(4) COMP.

  05 FILLER                  PIC X(38).

03 TEXT-LAST-MAINTAINED-DATA.
  05 TEXT-LAST-TIME         PIC 9(6).
  05 FILLER                 PIC XX.
  05 TEXT-LAST-DATE        PIC X(8).
  05 TEXT-LAST-TERMINID    PIC X(4).
  05 TEXT-LAST-USERID      PIC X(8).

03 TEXT-ADDED-MAINTAINED-DATA.
  05 TEXT-ADDED-TIME       PIC 9(6).
  05 FILLER                PIC XX.
  05 TEXT-ADDED-DATE       PIC X(8).
  05 TEXT-ADDED-TERMINID  PIC X(4).
  05 TEXT-ADDED-USERID    PIC X(8).

03 TEXT-DATA-AREA.
  05 FILLER                PIC X(1892).

*-----*
* - END -          *** COPYBOOK: MQICONFG ***          - END - *
*-----*

EJECT

*-----*
* ERROR HANDLEING
*-----*
01 WS-ERR.
* COPY          MQIERR.
*-----*
* - BEGIN -      *** COPYBOOK: MQIERR ***          - BEGIN - *
*-----*
* ERROR MODULE CALLING PARAMETERS
*-----*

02 ERR-HANDLER-COMMAREA.
05 ERR-CURRENT-INFO.
  10 ERR-COM-HANDLER       PIC X(48) VALUE SPACES.
  10 ERR-QUEUE             PIC X(48) VALUE SPACES.
  10 ERR-FILE              PIC X(8) VALUE SPACES.
  10 ERR-DETAIL            PIC X(80) VALUE SPACES.
  10 ERR-DETAIL2           PIC X(80) VALUE SPACES.
  10 ERR-Q-CODE            PIC S9(8) COMP VALUE ZERO.
  10 FILLER                PIC X(8) VALUE SPACES.

05 ERR-RESULTS.
  10 ERR-CODE              PIC 9(6) VALUE ZERO.
  10 FILLER                PIC XX VALUE SPACES.
  10 ERR-PROGRAM           PIC X(8) VALUE SPACES.
  10 ERR-TRANID            PIC X(4) VALUE SPACES.
  10 ERR-TERMINID         PIC X(4) VALUE SPACES.
  10 ERR-TASKNO            PIC S9(7) COMP-3 VALUE ZERO.
  10 ERR-ABSTIME           PIC S9(15) COMP-3 VALUE ZERO.

  10 ERR-DEBUG-EIBFN      PIC XX VALUE SPACES.
  10 ERR-DEBUG-EIBRCODE   PIC X(6) VALUE LOW-VALUES.
  10 ERR-DEBUG-EIBRSRCE   PIC X(8) VALUE LOW-VALUES.
  10 ERR-DEBUG-EIBRESP    PIC S9(8) COMP VALUE ZEROS.
  10 ERR-DEBUG-EIBRESP2   PIC S9(8) COMP VALUE ZEROS.
  10 ERR-DEBUG-EIBERRCD   PIC X(4) VALUE LOW-VALUES.
  10 ERR-DEBUG-ABEND      PIC X(4) VALUE SPACES.
  10 FILLER                PIC X(12) VALUE SPACES.

*-----*
* - END -          *** COPYBOOK: MQIERR ***          - END - *
*-----*

* COPY          MQIERRC.
*-----*
* IBM MQSERIES COMMON ERROR CODES
*-----*
01 MSG-ERROR-MESSAGES.
  05 ERR-NO-ENVIRONMENT    PIC 9(6) VALUE 900000.

  05 ERR-CICS-ERROR        PIC 9(6) VALUE 800000.
  05 ERR-CICS-INVALID-REQ  PIC 9(6) VALUE 800010.
  05 ERR-CICS-ILLOGIC      PIC 9(6) VALUE 800011.
  05 ERR-CICS-ERROR-CHECKPOINT PIC 9(6) VALUE 800090.
  05 ERR-CICS-ABEND        PIC 9(6) VALUE 800099.
  05 ERR-CICS-FILE-NOTOPEN PIC 9(6) VALUE 801012.
  05 ERR-CICS-DISABLE      PIC 9(6) VALUE 801019.
  05 ERR-CICS-NO-STORAGE   PIC 9(6) VALUE 802000.
  05 ERR-CICS-LENGTH-ERR  PIC 9(6) VALUE 803001.
  05 ERR-CICS-MAPFAIL      PIC 9(6) VALUE 808000.
  05 ERR-CICS-PGMIDERR     PIC 9(6) VALUE 809000.
  05 ERR-CICS-FILEID       PIC 9(6) VALUE 809010.
  05 ERR-CICS-NOFILE       PIC 9(6) VALUE 809011.
  05 ERR-CICS-IO-ERROR     PIC 9(6) VALUE 809012.
  05 ERR-CICS-TRANIDERR   PIC 9(6) VALUE 809050.

```

```

05 ERR-COM-FREE-ERROR          PIC 9(6) VALUE 501001.
05 ERR-COM-EIB-ERROR          PIC 9(6) VALUE 501002.
05 ERR-COM-STAT-ERROR         PIC 9(6) VALUE 501003.
05 ERR-COM-ALLOC-ERROR        PIC 9(6) VALUE 501004.
05 ERR-COM-ALLOC-RETRY        PIC 9(6) VALUE 501005.
05 ERR-COM-CONN-ERROR         PIC 9(6) VALUE 501006.
05 ERR-COM-SEND-ERROR         PIC 9(6) VALUE 501008.
05 ERR-COM-RECV-RESP-ERR     PIC 9(6) VALUE 501009.
05 ERR-COM-RESP-TYPE          PIC 9(6) VALUE 501010.
05 ERR-COM-RESP-MSN          PIC 9(6) VALUE 501011.
05 ERR-COM-RESP-FATAL        PIC 9(6) VALUE 501012.
05 ERR-COM-MSG-ERROR          PIC 9(6) VALUE 501013.
05 ERR-COM-BIG-INDIAN         PIC 9(6) VALUE 501014.
05 ERR-COM-TSH-ERROR          PIC 9(6) VALUE 501015.
05 ERR-COM-CCSID-ERROR        PIC 9(6) VALUE 501016.
05 ERR-COM-MSH-ERROR          PIC 9(6) VALUE 501017.
05 ERR-COM-MQX-ERROR          PIC 9(6) VALUE 501018.
05 ERR-COM-INIT-ERROR         PIC 9(6) VALUE 501019.
05 ERR-COM-FAP-ERROR          PIC 9(6) VALUE 501020.
05 ERR-COM-MSG-SIZE           PIC 9(6) VALUE 501021.
05 ERR-COM-WRAP-ERROR         PIC 9(6) VALUE 501022.
05 ERR-COM-MCP-DOWN           PIC 9(6) VALUE 501023.
05 ERR-COM-DOWN                PIC 9(6) VALUE 501024.
05 ERR-COM-NOT-FOUND          PIC 9(6) VALUE 501025.
05 ERR-COM-ERROR              PIC 9(6) VALUE 501026.
05 ERR-COM-BUSY               PIC 9(6) VALUE 501027.
05 ERR-COM-RESYNC-ERROR       PIC 9(6) VALUE 501028.
05 ERR-COM-STATUS-ERROR       PIC 9(6) VALUE 501029.
05 ERR-COM-LENGTH-ERROR      PIC 9(6) VALUE 501030.
05 ERR-COM-MSG-PER-BATCH      PIC 9(6) VALUE 501031.
05 ERR-COM-MAX-TRANSM-SIZE    PIC 9(6) VALUE 501032.
05 ERR-COM-RESET-MSN          PIC 9(6) VALUE 501050.

05 ERR-INT-LINK-ERROR         PIC 9(6) VALUE 400000.
05 ERR-INT-LINK-COM-SIZE      PIC 9(6) VALUE 400001.
05 ERR-INT-LINK-COM-DATA      PIC 9(6) VALUE 400002.
05 ERR-INT-RETURN-ERROR       PIC 9(6) VALUE 400003.
05 ERR-INT-MOVE-ERROR         PIC 9(6) VALUE 400010.
05 ERR-INT-STRUC-MISSING      PIC 9(6) VALUE 402000.
05 ERR-INT-STRUC-ERROR        PIC 9(6) VALUE 402090.

05 ERR-LOGIC-NOT-SUPPORTED    PIC 9(6) VALUE 300000.
05 ERR-LOGIC-STARTED-WRONG    PIC 9(6) VALUE 300010.
05 ERR-LOGIC-REPEATED-FAILURE PIC 9(6) VALUE 300020.
05 ERR-LOGIC-LOCKS-EXCEEDED   PIC 9(6) VALUE 300030.
05 ERR-LOGIC-MISSING-RECORD   PIC 9(6) VALUE 301000.
05 ERR-LOGIC-RECORD-DUPLICATED PIC 9(6) VALUE 301010.
05 ERR-LOGIC-Q-CKP-MISSING    PIC 9(6) VALUE 309010.

05 ERR-PROC-SYSTEM-STOPPED    PIC 9(6) VALUE 100000.
05 ERR-PROC-SYSTEM-ACTIVE     PIC 9(6) VALUE 100010.
05 ERR-PROC-SYS-START-NOQDR   PIC 9(6) VALUE 100011.
05 ERR-PROC-SYS-START-MAXQDR  PIC 9(6) VALUE 100012.
05 ERR-PROC-SYS-START-MAXCOM  PIC 9(6) VALUE 100013.
05 ERR-PROC-SYS-START-NOSYS   PIC 9(6) VALUE 100090.
05 ERR-PROC-Q-EXCEEDED-DEPTH  PIC 9(6) VALUE 101000.
05 ERR-PROC-Q-CONCURRENT-UPD  PIC 9(6) VALUE 101010.
05 ERR-PROC-Q-NOTFOUND        PIC 9(6) VALUE 101015.
05 ERR-PROC-Q-STOPPED         PIC 9(6) VALUE 101090.
05 ERR-PROC-Q-DISABLED        PIC 9(6) VALUE 101091.
05 ERR-PROC-QSN-LIMIT-REACHED PIC 9(6) VALUE 102090.
05 ERR-PROC-FILE-SPACE-PUT    PIC 9(6) VALUE 102091.
05 ERR-PROC-FILE-SPACE        PIC 9(6) VALUE 102092.
05 ERR-PROC-DUAL-Q-ERROR      PIC 9(6) VALUE 104021.
05 ERR-PROC-DUAL-Q-FILE       PIC 9(6) VALUE 104022.
05 ERR-PROC-DUAL-Q-LOGIC      PIC 9(6) VALUE 104023.
05 ERR-PROC-TRIGGER-ERROR     PIC 9(6) VALUE 105090.
05 ERR-PROC-TRIGGER-DATA      PIC 9(6) VALUE 105091.
05 ERR-PROC-NOT-AUTHORIZED    PIC 9(6) VALUE 109000.

05 ERR-WARN-SYS-STARTED-W-ERR PIC 9(6) VALUE 010000.
05 ERR-WARN-SYS-STARTED-W-FILER PIC 9(6) VALUE 010001.
05 ERR-WARN-SYS-STARTED-W-COMER PIC 9(6) VALUE 010002.
05 ERR-WARN-SYS-STARTED-W-CHANG PIC 9(6) VALUE 010003.
05 ERR-WARN-SYS-QUEUE-DISABLED PIC 9(6) VALUE 010005.
05 ERR-WARN-SYS-QUEUE-ENABLED PIC 9(6) VALUE 010007.

05 ERR-WARN-COM-CONNECT       PIC 9(6) VALUE 005000.
05 ERR-WARN-COM-OPENED        PIC 9(6) VALUE 005001.
05 ERR-WARN-COM-QUEUE-OPENED  PIC 9(6) VALUE 005002.
05 ERR-WARN-COM-LU62-CONNECT  PIC 9(6) VALUE 005003.

05 ERR-WARN-COM-RECEIVER-ALLOC PIC 9(6) VALUE 005004.
05 ERR-WARN-COM-QUEUE-EMPTY   PIC 9(6) VALUE 005005.
05 ERR-WARN-COM-QUEUE-CLOSED  PIC 9(6) VALUE 005006.
05 ERR-WARN-COM-DISC          PIC 9(6) VALUE 005007.
05 ERR-WARN-COM-SHUT          PIC 9(6) VALUE 005008.
05 ERR-WARN-COM-SHUT-SENT     PIC 9(6) VALUE 005009.

05 ERR-WARN-COM-TCP-CONNECT   PIC 9(6) VALUE 006003.
05 ERR-WARN-COM-TCP-STARTED  PIC 9(6) VALUE 006007.
05 TCP-FREE-ERROR             PIC 9(6) VALUE 006010.
05 TCP-CONN-ERROR             PIC 9(6) VALUE 006015.
05 TCP-SEND-ERROR             PIC 9(6) VALUE 006016.
05 TCP-RECV-RESP-ERROR        PIC 9(6) VALUE 006017.

05 ERR-FUNCTION-STARTED       PIC 9(6) VALUE 000100.
05 ERR-FUNCTION-DONE          PIC 9(6) VALUE 001000.
05 ERR-FUNCTION-NOT-DONE      PIC 9(6) VALUE 001090.

05 ERR-WARN-SYS-STARTED       PIC 9(6) VALUE 000000.

05 SYNCH-MSN-ERROR            PIC 9(6) VALUE 3.
05 SYNCH-MSG-DUP              PIC 9(6) VALUE 4.
05 ERR-WARN-COM-STARTED       PIC 9(6) VALUE 000007.
05 ERR-WRAP-COM-MISMATCH      PIC 9(6) VALUE 000009.
05 LU62-FREE-ERROR            PIC 9(6) VALUE 10.
05 LU62-EIB-ERROR             PIC 9(6) VALUE 11.
05 LU62-STAT-ERROR           PIC 9(6) VALUE 12.
05 LU62-ALLOC-ERROR          PIC 9(6) VALUE 13.
05 LU62-ALLOC-RETRY-ERROR    PIC 9(6) VALUE 14.
05 LU62-CONN-ERROR           PIC 9(6) VALUE 15.
05 LU62-SEND-ERROR           PIC 9(6) VALUE 16.
05 LU62-RECV-RESP-ERROR      PIC 9(6) VALUE 17.
05 INVLD-RESP-TYPE           PIC 9(6) VALUE 23.
05 INVLD-RESP-MSN            PIC 9(6) VALUE 24.
05 FATAL-RESP-TYPE           PIC 9(6) VALUE 25.
05 RECOVERABLE-RESP-TYPE     PIC 9(6) VALUE 26.
05 PARSE-MSN-ERROR           PIC 9(6) VALUE 29.
05 PARSE-TYPE-ERROR          PIC 9(6) VALUE 30.
05 PARSE-PDM-ERROR           PIC 9(6) VALUE 31.
05 PARSE-SID-ERROR           PIC 9(6) VALUE 32.
05 PARSE-PN-ERROR            PIC 9(6) VALUE 33.
05 PARSE-KEY-ERROR           PIC 9(6) VALUE 34.
05 PARSE-APID-ERROR          PIC 9(6) VALUE 35.
05 PARSE-ORG-DT-ERROR        PIC 9(6) VALUE 38.
05 PARSE-ORIG-MSN-ERROR      PIC 9(6) VALUE 39.
05 PARSE-BODY-ERROR          PIC 9(6) VALUE 40.
05 PARSE-STATUS-ERROR        PIC 9(6) VALUE 41.
05 PARSE-LENGTH-ERROR        PIC 9(6) VALUE 42.
05 MCONN-ERROR                PIC 9(6) VALUE 51.
05 MQOPEN-ERROR              PIC 9(6) VALUE 52.
05 MQGET-ERROR                PIC 9(6) VALUE 53.
05 MQPUT-ERROR                PIC 9(6) VALUE 54.
05 MQPT1-ERROR                PIC 9(6) VALUE 55.
05 MQCLOSE-ERROR             PIC 9(6) VALUE 56.
05 MQDISC-ERROR              PIC 9(6) VALUE 57.
05 QM-OTHER-ERROR            PIC 9(6) VALUE 60.
05 RECV-RETURN-LON-STATUS    PIC 9(6) VALUE 80.
05 RECV-RETURN-LON-TYPE      PIC 9(6) VALUE 81.
05 SIDRC-RETURN-MLP-FORMAT    PIC 9(6) VALUE 91.

*-----*
* COPY TTETST3.
*-----*
* DISPLAY MESSAGES FOR TTPTST3
*-----*
* NORMAL MESSAGES
01 MSG-NORMAL.
05 MSG-START                   PIC X(60) VALUE
    'ENTER START VALUES.'.
05 MSG-END                     PIC X(60) VALUE
    'TEST3 HAS ENDED.'.
05 MSG-OK                       PIC X(60) VALUE
    'FUNCTION COMPLETED - ENTER NEW REQUEST.'.
05 MSG-RETURNING                PIC X(60) VALUE
    'FUNCTION COMPLETED - ENTER NEW REQUEST.'.

05 MSG-SYSTEM-INACTIVE          PIC X(60) VALUE
    'QUEUING SYSTEM IS NOT ACTIVE'.

* ERROR MESSAGES
01 MSG-ERROR.

```

```

05 MSG-ERR-QUEUE          PIC X(60) VALUE
'QUEUE NAME NOT ENTERED.'.
05 MSG-ERR-TS             PIC X(60) VALUE
'TIME STAMP FLAG MUST BE SPACE OR Y.'.
05 MSG-ERR-MSG           PIC X(60) VALUE
'TEXT MESSAGE NOT ENTERED.'.
05 MSG-ERR-MSG-SIZE      PIC X(60) VALUE
'TEXT MESSAGE SIZE NOT ENTERED.'.
05 MSG-ERR-MSG-SIZE-VALUE PIC X(60) VALUE
'TEXT MESSAGE SIZE IF INVALID.'.
05 MSG-ERR-NUM-MSG       PIC X(60) VALUE
'NUMBER OF MESSAGES TO BE PUT PER TASK NOT ENTERED.'.
05 MSG-ERR-NUM-MSG-VALUE PIC X(60) VALUE
'NUMBER OF MESSAGES TO BE PUT PER TASK IS INVALID.'.
05 MSG-ERR-MAX-TASK      PIC X(60) VALUE
'NUMBER OF TASKS TO START NOT ENTERED.'.
05 MSG-ERR-MAX-TASK-VALUE PIC X(60) VALUE
'NUMBER OF TASKS TO START IS INVALID.'.

05 MSG-ERR-FUNCTION      PIC X(60) VALUE
'FUNCTION NOT ENTERED.'.
05 MSG-ERR-FUNCTION-VALUE PIC X(60) VALUE
'FUNCTION MUST BE A "G" OR "P".'.

05 MSG-ERR-PFKEY         PIC X(60) VALUE
'INVALID PFKEY WAS ENTERED - ENTER VALID ONE.'.
05 MSG-ERR-MAPFAIL      PIC X(60) VALUE
'TASK ENTERED IMPROPERLY - TASK RE-STARTED.'.

05 MSG-ERR-MAPFAIL-REPEATED PIC X(60) VALUE
'TASK HAS REPEATED ERRORS - PLEASE CONTACT SUPPORT.'.

*--MAJOR ERROR THAT ARE LOGGED
05 MSG-ERR-CICS          PIC X(60) VALUE
'CICS ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-TRANS-ID     PIC X(60) VALUE
'OPTION NOT AVAILABLE- PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-NOFILE       PIC X(60) VALUE
'CICS FILE ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-DISABLED     PIC X(60) VALUE
'CICS DISABLE ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-ILLOGIC      PIC X(60) VALUE
'CICS ILLOGIC ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-INVREQ       PIC X(60) VALUE
'CICS REQUEST ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-IOERR        PIC X(60) VALUE
'CICS I/O ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-NOTFOUND     PIC X(60) VALUE
'CICS NOTFOUND ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-NOTOPEN      PIC X(60) VALUE
'CICS NOTOPEN ERROR - PLEASE CONTACT SUPPORT.'.
05 MSG-ERR-ABENDED      PIC X(60) VALUE
'CICS ABEND ERROR - PLEASE CONTACT SUPPORT.'.

05 MSG-ERR-USER-NOT-AUTH PIC X(60) VALUE
'USER IS NOT AUTHORIZED TO PERFORM FUNCTION.'.

*-----*
* COPY      MQWEOWS.
*-----*
*Debugging eyecatcher information for end of WORKING-STORAGE.
*-----*
01 FILLER          PIC X(16) VALUE
'====='.
01 WS-RWS-PROGRAM-NAME2 PIC X(8).
01 FILLER          PIC X(16) VALUE
' Version V2.1.0'.
*-----*
*-----*
LINKAGE SECTION.
*-----*
01 DFHCOMMAREA.
05 FILLER          PIC X(400).

*--STARTED DATA
01 LK-GET-DATA.
05 FILLER          PIC X(400).

*-----*
PROCEDURE DIVISION.
-----*
0000-MAIN.
*--SETUP ENVIRONMENT FROM LAST TIME
PERFORM 1000-INITIAL.

*-- --IF RECIEVEING - PROCESS FUNCTION
IF MTP-SCREEN-RECEIVE
THEN
PERFORM 2000-SCREEN-FUNCTION
THRU 2000-SCREEN-EXIT.
*
*-----*
0000-RETURN-MQMS.
PERFORM 7000-SEND-MAP.
MOVE 'R' TO MTP-SCREEN-IND.
*
EXEC CICS RETURN TRANSID(MTP-ACTIVE-TASK)
COMMAREA(MTP-COMMAREA)
LENGTH (LENGTH OF MTP-COMMAREA)
END-EXEC.
*
GOBACK.
EJECT
*-----*
1000-INITIAL.
*-----*
* PURPOSE: SETUP HANDLES
* CHECK IF ENVIRONMENT EXIST - ALREADY
* IF FIRST TIME - JUST SET MAIN SCREEN AND GET OUT
*-----*
EXEC CICS HANDLE CONDITION
ERROR (9900-HANDLE-ERROR)
TRANSIDERR (9900-HANDLE-TRANSID)
MAPFAIL (9900-HANDLE-MAPFAIL)
FILENOTFOUND (9900-HANDLE-NOFILE)
DISABLED (9900-HANDLE-DISABLE)
ILLOGIC (9900-HANDLE-ILLOGIC)
INVREQ (9900-HANDLE-INVREQ)
IOERR (9900-HANDLE-IOERR)
NOTFND (9900-HANDLE-NOTFOUND)
NOTOPEN (9900-HANDLE-NOTOPEN)
END-EXEC.

*--SET ERROR INFO
PERFORM 1050-SET-ERROR-INFO.

*--GET WHAT SYSTEM / APPLIC IS RUNNING
EXEC CICS ASSIGN SYSID (WS-SYSID)
APPLID (WS-APPLID)
STARTCODE (WS-STARTCD)
END-EXEC.

*--CHECK IF SYSTEM EXIST - ALREADY
PERFORM 1100-CHECK-SYSTEM
THRU 1100-EXIT.

*--SETUP ENVIRONMENT
PERFORM 1200-SETUP-ENVIR
THRU 1200-EXIT.

*-----*
1000-EXIT.
EXIT.
EJECT
*-----*
1050-SET-ERROR-INFO.
*-----*
* PURPOSE: SET DEFAULT ERROR INFO
*-----*
*--SET CSMT DATE AND TIME
EXEC CICS ASKTIME
ABSTIME(WS-ABSTIME)
END-EXEC.

MOVE EIBTIME TO WS-UNPACK-TIME-9.
MOVE WS-TIME-HH TO WS-FORMAT-TIME-HH
MOVE WS-TIME-MM TO WS-FORMAT-TIME-MM.
MOVE WS-TIME-SS TO WS-FORMAT-TIME-SS.

EXEC CICS FORMATTIME

```

```

                ABSTIME (WS-ABSTIME)
                MMDDYY (WS-FORMATTED-DATE)
                DATESEP ('/')
            END-EXEC.
*-- Save the year field which is held in the century field
* immediately after the FORMATTIME
  MOVE WS-FORMAT-DATE-CC TO WS-FORMAT-DATE-YY
*
  EXEC CICS FORMATTIME
        ABSTIME(WS-ABSTIME)
        YYMMDD (WS-DATE-YYMMDD)
  END-EXEC.

*-- --SET CENTURY
  IF WS-DATE-YY > 60
    THEN
      MOVE 19          TO WS-DATE-CC
                       WS-FORMAT-DATE-CC
    ELSE
      MOVE 20          TO WS-DATE-CC
                       WS-FORMAT-DATE-CC
  END-IF

*--SET COMMON ERROR INFO
  MOVE ZERO          TO ERR-CODE.
  MOVE 'TTPTST3'    TO ERR-PROGRAM.

*-----*
  EJECT
*-----*
  1100-CHECK-SYSTEM.
*-----*
* PURPOSE: LINK TO FINQ TO GET SYSTEM STATUS
*-----*
*--SET UP COMMAREA
  MOVE SPACES        TO FINDQ-CALL-PARAMETERS.
  MOVE 'S'           TO FINDQ-CALL-TYPE.

*--CALL
  EXEC CICS LINK PROGRAM (ENV-II-LINK-FINDQ)
                  COMMAREA (FINDQ-CALL-PARAMETERS)
                  LENGTH(LENGTH OF FINDQ-CALL-PARAMETERS)
  END-EXEC.
*
*-----*
  1100-EXIT.
  EXIT.
  EJECT
*-----*
  1200-SETUP-ENVIR.
*-----*
* PURPOSE: SETUP PROGRAM ENVIR
*-----*
*--SETUP NEW COMMON AREA
  MOVE LOW-VALUES TO MAINO.

*--IF NOT RE-STARTED
  IF NOT WS-STARTED
    THEN
*-- --IF NOT STARTED AND NO COMMAREA - JUST SETUP TO MAIN...
  IF (EIBCALEN EQUAL ZERO)
    THEN
      MOVE 'S'          TO MTP-SCREEN-IND
      MOVE MSG-START    TO WS-ERROR-MESSAGE
    ELSE
*-- --MOVE COMMAREA TO WORKING-STORAGE..CONTINUE
      MOVE DFHCOMMAREA TO MTP-COMMAREA
      MOVE 'R'          TO MTP-SCREEN-IND
  END-IF.

*--STARTED - TREAT AS NEW TASK
  IF WS-STARTED
    THEN
      PERFORM 1210-GET-STARTED-DATA
      THRU 1210-GET-STARTED-EXIT

*-- --IF RETURNING FROM ANOTHER APPLI. - TREAT AS NEW
  MOVE LOW-VALUES TO MAINO
  IF MTP-SCREEN-RETURN
    THEN
      MOVE MSG-RETURNING TO WS-ERROR-MESSAGE
      MOVE MTP-CONFIG-DATA

                MOVE SPACES          TO MTP-MAIN-TASK
                MOVE SPACES          TO MTP-CONFIG-DATA
            ELSE
                MOVE MSG-START      TO WS-ERROR-MESSAGE
            END-IF
            MOVE 'S'                TO MTP-SCREEN-IND.

*--SETUP TASK ID
  MOVE EIBTRNID TO MTP-ACTIVE-TASK.

*-----*
  1200-EXIT.
  EXIT.
  EJECT
*-----*
  1210-GET-STARTED-DATA.
*-----*
* PURPOSE: READ STARTED DATA
*-----*
*--GET
  EXEC CICS RETRIEVE
        SET (ADDRESS OF LK-GET-DATA)
        LENGTH (WS-REC-SIZE)
  END-EXEC.
*
  IF WS-REC-SIZE NOT < LENGTH OF MTP-COMMAREA
    THEN
*-- --GOT VALID LENGTH- MOVE AND CHECK
      MOVE LK-GET-DATA TO MTP-COMMAREA
      IF NOT MTP-HEADER-OK
*-- --ERROR IN GET DATA - RESET COMMAREA
      THEN
          MOVE SPACES TO MTP-COMMAREA
          SET MTP-HEADER-OK TO TRUE
          MOVE 'S' TO MTP-SCREEN-IND
          MOVE 'MAIN' TO MTP-MAP-VALUE.
*
*-----*
  1210-GET-STARTED-EXIT.
  EXIT.
  EJECT
*-----*
  2000-SCREEN-FUNCTION.
*-----*
* PURPOSE: GET MAIN MAP
* CHECK OPTION KEYS
* CHECK OPTION FIELD
* PROCESS FUNCTION ENTERED
*-----*
*--PRELIMINARY EDIT OF PF KEYS
  PERFORM 2100-MAIN-CHECK-KEYS.
  IF NOT WS-EDIT-ERR
    THEN
*--GET MAP
      PERFORM 7000-RECEIVE-MAP

*--IF RECORD NOT FOUND - SET UP DEFAULT RECORD
  IF NOT FINDQ-SYSTEM-ACTIVE
    THEN
      MOVE MSG-SYSTEM-INACTIVE
      TO WS-ERROR-MESSAGE
    ELSE
*-- --EDIT MAP
      PERFORM 2200-MAIN-EDIT
      THRU 2200-MAIN-EXIT

*--PROCESS FUNCTION KEY - IF NO ERRORS
  IF NOT WS-EDIT-ERR
    THEN
      PERFORM 2300-MAIN-FUNCTION
      THRU 2300-MAIN-EXIT.

*-----*
  2000-SCREEN-EXIT.
  EXIT.
  EJECT
*-----*
  2100-MAIN-CHECK-KEYS.
*-----*
* PURPOSE: PRELIMINARY PF KEY CHECK
*-----*

```

```

*--CHECK AID KEY
*-- --MAIN MENU
  IF (EIBAID EQUAL DFHPF2)
  AND (MTP-MAIN-TASK NOT EQUAL SPACES)
  THEN
    GO TO 9000-MAIN-MENU.

*-- --SHUTDOWN
  IF ((EIBAID EQUAL DFHCLEAR OR DFHPA1 OR DFHPA2)
  OR (EIBAID EQUAL DFHPF3))
  THEN
    GO TO 9000-SHUTDOWN.

*-- --QUEUE KEYS - FIRST INQ THEN UPDATE
  IF (EIBAID EQUAL DFHPF4)
  OR (EIBAID EQUAL DFHENTER)
  THEN
    NEXT SENTENCE
  ELSE
    MOVE -1 TO LTNUML
    MOVE 'Y' TO WS-EDIT-ERR-FLAG
    MOVE MSG-ERR-PFKEY TO WS-ERROR-MESSAGE.

*--SET TYPE OF FUNCTION - DEFAULT TO UPDATE
  MOVE 'UPDATE' TO MTP-MAP-FUNCTION.
*
*-----*
  EJECT
*-----*
  2200-MAIN-EDIT.
*-----*
* PURPOSE: EDIT SCREEN
*-----*

*--FUNCTION
  MOVE DFHBMFSE TO LFUNCA.
  IF (LFUNCI EQUAL '?')
  OR (LFUNCI NOT > SPACE)
  THEN
    MOVE '?' TO LFUNCO
    MOVE -1 TO LFUNCL
    MOVE DFHUNIMD TO LFUNCA
    MOVE 'Y' TO WS-EDIT-ERR-FLAG
    MOVE MSG-ERR-FUNCTION
    TO WS-ERROR-MESSAGE
    PERFORM 8000-MOVE-ERR-MESSAGE
  ELSE
    IF (LFUNCI EQUAL 'P')
    THEN
      MOVE 'PUT' TO TST2-FUNCTION
    ELSE
      IF (LFUNCI EQUAL 'G')
      THEN
        MOVE 'GET' TO TST2-FUNCTION
      ELSE
        MOVE -1 TO LFUNCL
        MOVE DFHUNIMD TO LFUNCA
        MOVE 'Y' TO WS-EDIT-ERR-FLAG
        MOVE MSG-ERR-FUNCTION-VALUE
        TO WS-ERROR-MESSAGE
        PERFORM 8000-MOVE-ERR-MESSAGE.

*--NUMBER OF STARTS
  MOVE DFHBMFSE TO LTNUMA.
  IF (LTNUMI EQUAL '?')
  OR (LTNUMI NOT > SPACE)
  THEN
    MOVE '?' TO LTNUMO
    MOVE -1 TO LTNUML
    MOVE DFHUNIMD TO LTNUMA
    MOVE 'Y' TO WS-EDIT-ERR-FLAG
    MOVE MSG-ERR-MAX-TASK
    TO WS-ERROR-MESSAGE
    PERFORM 8000-MOVE-ERR-MESSAGE
  ELSE
    IF (LTNUMI NUMERIC)
    THEN
      MOVE LTNUMI TO WS-NUM
      IF (WS-NUM < 0)
      THEN
        MOVE -1 TO LTNUML

*--CHECK QUEUE FIELD
  MOVE DFHBMFSE TO LPQUEA.
  IF (LPQUEI EQUAL '?')
  OR (LPQUEI NOT > SPACE)
  THEN
    MOVE '?' TO LPQUEO
    MOVE -1 TO LPQUEL
    MOVE DFHUNIMD TO LPQUEA
    MOVE 'Y' TO WS-EDIT-ERR-FLAG
    MOVE MSG-ERR-QUEUE
    TO WS-ERROR-MESSAGE
    PERFORM 8000-MOVE-ERR-MESSAGE
  ELSE
    MOVE LPQUEI TO TST2-PUT-QUEUE-NAME.

*--NUM OF MESSAGE PER TASK
  MOVE DFHBMFSE TO LMNUMA.
  IF (LMNUMI EQUAL '?')
  OR (LMNUMI NOT > SPACE)
  THEN
    MOVE '?' TO LMNUMO
    MOVE -1 TO LMNUML
    MOVE DFHUNIMD TO LMNUMA
    MOVE 'Y' TO WS-EDIT-ERR-FLAG
    MOVE MSG-ERR-NUM-MSG
    TO WS-ERROR-MESSAGE
    PERFORM 8000-MOVE-ERR-MESSAGE
  ELSE
    IF (LMNUMI NUMERIC)
    THEN
      MOVE LMNUMI TO WS-NUM
      IF (WS-NUM < 0)
      THEN
        MOVE -1 TO LMNUML
        MOVE DFHUNIMD TO LMNUMA
        MOVE 'Y' TO WS-EDIT-ERR-FLAG
        MOVE MSG-ERR-NUM-MSG-VALUE
        TO WS-ERROR-MESSAGE
        PERFORM 8000-MOVE-ERR-MESSAGE
      ELSE
        MOVE WS-NUM TO TST2-PUT-NUM-MSG
    ELSE
      MOVE -1 TO LMNUML
      MOVE DFHUNIMD TO LMNUMA
      MOVE 'Y' TO WS-EDIT-ERR-FLAG
      MOVE MSG-ERR-NUM-MSG-VALUE
      TO WS-ERROR-MESSAGE
      PERFORM 8000-MOVE-ERR-MESSAGE.

*--MESSAGE SIZE
  MOVE DFHBMFSE TO LPSIZEA.
  IF TST2-FUNCT-PUT
  THEN
    IF ((LPSIZEI EQUAL '?') OR (LPSIZEI NOT > SPACE))
    THEN
      MOVE '?' TO LPSIZEO
      MOVE -1 TO LPSIZEL
      MOVE DFHUNIMD TO LPSIZEA
      MOVE 'Y' TO WS-EDIT-ERR-FLAG
      MOVE MSG-ERR-MSG-SIZE
      TO WS-ERROR-MESSAGE
      PERFORM 8000-MOVE-ERR-MESSAGE
    ELSE
      IF (LPSIZEI NUMERIC)
      THEN
        MOVE LPSIZEI TO WS-NUM

```

```

IF (WS-NUM < 0) OR
(WS-NUM > 32000)
THEN
MOVE -1 TO LPSIZEL
MOVE DFHUNIMD TO LPSIZEA
MOVE 'Y' TO WS-EDIT-ERR-FLAG
MOVE MSG-ERR-MSG-SIZE-VALUE
TO WS-ERROR-MESSAGE
PERFORM 8000-MOVE-ERR-MESSAGE
ELSE
MOVE WS-NUM TO TST2-PUT-MSG-SIZE
ELSE
MOVE -1 TO LPSIZEL
MOVE DFHUNIMD TO LPSIZEA
MOVE 'Y' TO WS-EDIT-ERR-FLAG
MOVE MSG-ERR-MSG-SIZE-VALUE
TO WS-ERROR-MESSAGE
PERFORM 8000-MOVE-ERR-MESSAGE.

*--CHECK MESSAGE
MOVE DFHBMFSE TO LMSGA.
IF TST2-FUNCT-PUT
THEN
IF (LMSGI EQUAL '?' ) OR (LMSGI NOT > SPACE)
THEN
MOVE '?' TO LMSGO
MOVE -1 TO LMSGL
MOVE DFHUNIMD TO LMSGA
MOVE 'Y' TO WS-EDIT-ERR-FLAG
MOVE MSG-ERR-MSG
TO WS-ERROR-MESSAGE
PERFORM 8000-MOVE-ERR-MESSAGE
ELSE
MOVE LMSGI TO TST2-PUT-MSG.

*--CHECK TIME STAMP FLAG
MOVE DFHBMFSE TO LTSA.
IF TST2-FUNCT-PUT
THEN
IF (LTSI EQUAL '?' ) OR (LTSI NOT > SPACE)
THEN
MOVE '?' TO LTSO
MOVE -1 TO LTSL
MOVE DFHUNIMD TO LTSA
MOVE 'Y' TO WS-EDIT-ERR-FLAG
MOVE MSG-ERR-TS
TO WS-ERROR-MESSAGE
PERFORM 8000-MOVE-ERR-MESSAGE
ELSE
IF TST2-PUT-MSG-TIMESTAMP EQUAL SPACE OR 'Y'
THEN
MOVE LTSI TO TST2-PUT-MSG-TIMESTAMP.

-----*
2200-MAIN-EXIT.
EXIT.
EJECT
-----*
2300-MAIN-FUNCTION.
-----*
* PURPOSE: SETUP DEFAULT RECORD AND MESSAGE
* DEFAULT TO QUEUE PROCESSING
-----*
*--SET CURSOR
MOVE -1 TO LTNUML.

*--START TASK.
PERFORM WS-SS-STARTS TIMES
EXEC CICS START TRANSID('TST2')
INTERVAL (000000)
FROM (WS-TST2-COMMAREA)
LENGTH (LENGTH OF WS-TST2-COMMAREA)
END-EXEC
END-PERFORM.

*--SAYS OK
MOVE MSG-OK TO WS-ERROR-MESSAGE.

-----*
2300-MAIN-EXIT.
EXIT.

EJECT
EJECT
-----*
7000-RECEIVE-MAP.
-----*
* PURPOSE: GET USER MAP
-----*
EXEC CICS RECEIVE MAP (MTP-MAP-VALUE)
MAPSET('TTMTST3')
INTO (MAIN0)
END-EXEC.
*
-----*
EJECT
-----*
7000-SEND-MAP.
-----*
* PURPOSE: SETUP HEADER DATA
* SEND SCREEN BASED ON MODE
-----*
*--SETUP HEADER
PERFORM 7100-SETUP-HEADER.

*--RESET ERROR TO FIRST ONE..IF MORE THAN ONE
IF MTP-ERR-COUNT > ZERO
THEN
MOVE WS-ERR-MSG (1)
TO WS-ERROR-MESSAGE.
*
*--SEND SCREEN
IF MTP-SCREEN-SEND
THEN
*-- --NEW MAP - SETUP INFO....
MOVE WS-ERROR-MESSAGE TO LERRO
EXEC CICS SEND MAP (MTP-MAP-VALUE)
MAPSET('TTMTST3')
FROM (MAIN0)
ERASE CURSOR
END-EXEC
ELSE
MOVE WS-ERROR-MESSAGE TO LERRO
EXEC CICS SEND MAP (MTP-MAP-VALUE)
MAPSET('TTMTST3')
FROM (MAIN0)
DATAONLY CURSOR
END-EXEC.

-----*
EJECT
-----*
7100-SETUP-HEADER.
-----*
* PURPOSE: SETUP HEADER DATA
-----*
*--SETUP HEADER
MOVE WS-FORMATTED-DATE TO MDATELO.
MOVE DFHBMPRF TO MDATELA.
MOVE WS-FORMATTED-TIME TO MTIMELO.

MOVE WS-SYSID TO MSYSTLO.
MOVE EIBTRMID TO MTERMLO.
MOVE WS-APPLID TO MAPPLLO.

-----*
EJECT
-----*
8000-MOVE-ERR-MESSAGE.
-----*
* PURPOSE: MOVE MULTIPLE ERROR MESSAGES...
-----*
ADD +1 TO WS-ERR-COUNT.
IF WS-ERR-COUNT NOT > WS-ERR-MAX
THEN
MOVE WS-ERROR-MESSAGE
TO WS-ERR-MSG (WS-ERR-COUNT).

-----*
EJECT
-----*
9000-SHUTDOWN.
-----*

```

TTPTST3.Z

```

* PURPOSE: SHUTDOWN PROGRAM
*-----*
*--IF ORIGIN TRAN WAS ME .....
  EXEC CICS SEND FROM (MSG-END)
    LENGTH (LENGTH OF MSG-END) ERASE
    END-EXEC.
*
  EXEC CICS RETURN
    END-EXEC.
*-----*
  EJECT
*-----*
9900-MAIN-MENU.
*-----*
* PURPOSE: RETURN TO MAIN TASK
*-----*
*--RE-START ORIGINAL TASK
  MOVE SPACE TO MTP-SCREEN-IND.
  EXEC CICS START TRANSID(MTP-MAIN-TASK)
    TERMID(EIBTRMID)
    FROM (MTP-COMMAREA)
    LENGTH(LENGTH OF MTP-COMMAREA)
    INTERVAL(0)
    NOHANDLE
    END-EXEC.
*
  EXEC CICS RETURN
    END-EXEC.
*-----*
  EJECT
*-----*
* PURPOSE: ENVIRONMENT NOT SETUP
*-----*
9900-NO-ENVIR-SETUP.
  EXEC CICS SEND FROM (ENV-IT-UN-INIT-MSG)
    LENGTH (LENGTH OF ENV-IT-UN-INIT-MSG) ERASE
    END-EXEC
  EXEC CICS RETURN
    END-EXEC.
  EJECT
*-----*
* PURPOSE: ERROR CONDITION
*-----*
9900-HANDLE-TRANSID.
  MOVE ERR-CICS-TRANIDERR TO ERR-CODE.
  MOVE WS-TRAN-ID TO ERR-DETAIL.
  MOVE MSG-ERR-TRANS-ID TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-NOTAUTH.
  MOVE ERR-PROC-NOT-AUTHORIZED TO ERR-CODE.
  MOVE WS-TRAN-ID TO ERR-DETAIL.
  MOVE MSG-ERR-USER-NOT-AUTH TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-ERROR.
  MOVE ERR-CICS-ERROR TO ERR-CODE.
  MOVE MSG-ERR-CICS TO WS-ERROR-MESSAGE.
  GO TO 9999-FATAL-ERR-EXIT.
9900-HANDLE-NOFILE.
  MOVE ERR-CICS-NOFILE TO ERR-CODE.
  MOVE MSG-ERR-NOFILE TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-DISABLE.
  MOVE ERR-CICS-DISABLE TO ERR-CODE.
  MOVE MSG-ERR-DISABLED TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-ILLOGIC.
  MOVE ERR-CICS-ILLOGIC TO ERR-CODE.
  MOVE MSG-ERR-ILLOGIC TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-INVREQ.
  MOVE ERR-CICS-INVALID-REQ TO ERR-CODE.
  MOVE MSG-ERR-INVREQ TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-IOERR.
  MOVE ERR-CICS-IO-ERROR TO ERR-CODE.
  MOVE MSG-ERR-IOERR TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-NOTFOUND.
  MOVE ERR-LOGIC-MISSING-RECORD TO ERR-CODE.
  MOVE MSG-ERR-NOTFOUND TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-NOTOPEN.
  MOVE ERR-CICS-FILE-NOTOPEN TO ERR-CODE.
  MOVE MSG-ERR-NOTOPEN TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
9900-HANDLE-MAPFAIL.
  EXEC CICS HANDLE CONDITION
    MAPFAIL (9999-FATAL-ERR-PRE-EXIT)
    END-EXEC.
  MOVE ERR-CICS-MAPFAIL TO ERR-CODE.
  MOVE MSG-ERR-MAPFAIL TO WS-ERROR-MESSAGE.
  GO TO 9900-ERR-EXIT.
  EJECT
*-----*
9900-ERR-EXIT.
*-----*
* PURPOSE: ERROR CONDITION
* SEND SCREEN
* GO TO CICS RETURN W/ NEXT TRAN ID
*-----*
*--TRANSLATE ERROR CODE
  PERFORM 9999-CONVERT-ERROR-INFO.
*--WRITE ERROR MESSAGE
  PERFORM 9999-ERROR-WRITE.
*--RE-SEND MAIN MAP
  MOVE LOW-VALUES TO MAINO.
  MOVE -1 TO LTNULM.
  MOVE 'F' TO MTP-SCREEN-IND.
  GO TO 0000-RETURN-MQMS.
  EJECT
*-----*
9999-FATAL-ERR-PRE-EXIT.
*-----*
* PURPOSE: REPEATED MAPFAIL
*-----*
*--SET ERROR MESSAGE
  MOVE MSG-ERR-MAPFAIL-REPEATED TO WS-ERROR-MESSAGE.
  GO TO 9999-FATAL-ERR-EXIT.
*-----*
9999-FATAL-ERR-EXIT.
*-----*
* PURPOSE: ERROR EXIT - FOR REPEATED MAPFAIL / ABEND
*-----*
*--SEND MESSAGE
  EXEC CICS SEND FROM (WS-ERROR-MESSAGE)
    LENGTH (LENGTH OF WS-ERROR-MESSAGE) ERASE NOHANDLE
    END-EXEC.
*--GET OUT
  EXEC CICS RETURN
    END-EXEC.
  EJECT
*-----*
* ERROR HANDLING CODE
*-----*
* COPY MQIERRCD.
/INCLUDE MQIERRCD
*--ABEND MESSAGE SENT....JUST GET OUT
  MOVE MSG-ERR-ABENDED TO WS-ERROR-MESSAGE.
  GO TO 9999-FATAL-ERR-EXIT.

```


Sample program MQPECHO.Z

```

COPY COPYRSAP.
IDENTIFICATION DIVISION.
PROGRAM-ID.    MQPECHO.
AUTHOR.       IBM.
*****
*           T E S T      E C H O           *
*                                           *
*           A P P L I C A T I O N      I N T E R F A C E           *
*                                           *
*           M Q S e r i e s   f o r   V S E / E S A                   *
*-----*
* MQPECHO - IBM APPLICATION TEST PROGRAM TO ECHO MESSAGE FROM *
* A QUEUE, SAY XXX, TO A QUEUE NAMED 'IBM.REPLY.QUEUE' *
*-----*
* PREREQUISITE: *
* 1. SENDING QUEUE, A LOCAL QUEUE NAMED XXX, MUST BE *
*    DEFINED WITH *
*    TRIGGER ENAABLE: Y *
*    PROGRAM ID : MQPECHO *
* 2. SENDING QUEUE MUST BE ABLE TO TRIGGER MQPECHO *
*    A. IF XXX HAS MESSAGES, STOP THEN START XXX *
*    B. IF XXX DOESN'T HAVE ANY MESSAGES OR YOU WANT TO *
*       ECHO MORE MESSAGES THAN EXISTING ONES, THEN PUT *
*       SOME MESSAGES BY, EG, TST1 PUT 99 XXX *
* 3. DEFINE IBM.REPLY.QUEUE, IF IT DOES NOT EXIST *
*-----*
* FUNCTIONS: 1. ACTIVATED VIA TRIGGER MECHANISM BY QUEUE XXX. *
*            2. READ QUEUE XXX TILL THERE IS NO MORE MESSAGE *
*            3. ECHO READ MESSAGE INTO IBM.REPLY.QUEUE *
*-----*
* COPYBOOKS: MQIVALUE - IBM RETURN CODES. *
*            MQIERR - ERROR COMMAREA *
*            MQIERRC - ERROR COMMON CODES *
*            MQIERRCD - ERROR CODE *
*            MQICENV - ENVIRONMENT *
*-----*
* CALLS : MQCONN - CONNECT *
*         MQOPEN - OPEN *
*         MQPUT - PUT *
*         MQGET - GET *
*         MQCLOSE - CLOSE *
*         MQDISC - DISCONNECT *
*-----*
* CALLED BY: -- NONE -- *
*-----*
* CHANGE SUMMARY: *
*-----*
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
DATA DIVISION.
*****
WORKING-STORAGE SECTION.
* COPY COPYRWS.
*-----*
* COPYRIGHT WORKING STORAGE FOR COBOL MODULES *
*-----*
01 FILLER.
   05 FILLER PIC X(72) VALUE
      'Licensed Materials - Property of IBM'.
   05 FILLER PIC X(72) VALUE SPACES.
   05 FILLER PIC X(72) VALUE
      '5686-A06 '.
   05 FILLER PIC X(72) VALUE SPACES.
   05 FILLER PIC X(72) VALUE
      '(C) Copyright IBM Corp. 1998 All Rights Reserve
      'd'.
   05 FILLER PIC X(72) VALUE SPACES.
   05 FILLER PIC X(72) VALUE
      'US Government Users Restricted Rights - Use, duplication or
      ' '.
   05 FILLER PIC X(72) VALUE
      'disclosure restricted by GSA ADP Schedule Contract with IBM
      ' Corp.'.
*-----*
* Debugging eyecatcher information for start of WORKING-STORAGE.
*-----*
01 WS-RWS-PROGRAM-NAME1 PIC X(8).
01 FILLER PIC X(16) VALUE
   ' Version V2.1.0'.
01 WS-RWS-WHEN-COMPILED PIC X(21).
01 FILLER PIC X(7) VALUE '====='.
*-----*
*-----*
COPY MQWTRACE.
01 WS-WORK-FIELDS.
   05 WS-MORE-FLAG PIC XX VALUE SPACES.
   88 WS-MORE-DATA VALUE SPACES.
   88 WS-NOMORE-DATA VALUE 'Y'.
   05 WS-DATA-LENGTH PIC S9(4) COMP VALUE ZERO.
   05 WS-APPL-MSG-LENGTH PIC S9(8) COMP VALUE ZERO.
   05 WS-ABSTIME PIC S9(15) COMP-3.
   05 WS-DATE.
      10 WS-DATE-CC PIC 99 VALUE ZERO.
      10 WS-DATE-YMMMDD.
         12 WS-DATE-YY PIC 99 VALUE ZERO.
         12 WS-DATE-MM PIC 99 VALUE ZERO.
         12 WS-DATE-DD PIC 99 VALUE ZERO.
         12 FILLER PIC XX VALUE ZERO.
   05 WS-UNPACK-TIME-9 PIC 9(07) VALUE ZEROS.
   05 WS-UNPACK-TIME-X REDEFINES WS-UNPACK-TIME-9.
      10 FILLER PIC X(01).
      10 WS-TIME-HHMMSS.
         12 WS-TIME-HH PIC X(02).
         12 WS-TIME-MM PIC X(02).
         12 WS-TIME-SS PIC X(02).
   05 WS-FORMATTED-TIME.
      10 WS-FORMAT-TIME-HH PIC X(02) VALUE SPACES.
      10 FILLER PIC X(01) VALUE ':'.
      10 WS-FORMAT-TIME-MM PIC X(02) VALUE SPACES.
      10 FILLER PIC X(01) VALUE ':'.
      10 WS-FORMAT-TIME-SS PIC X(02) VALUE SPACES.
   05 WS-FORMATTED-DATE.
      10 WS-FORMAT-DATE-MM PIC X(02) VALUE SPACES.
      10 FILLER PIC X(01) VALUE '/'.
      10 WS-FORMAT-DATE-DD PIC X(02) VALUE SPACES.
      10 FILLER PIC X(01) VALUE '/'.
      10 WS-FORMAT-DATE-YY PIC X(02) VALUE SPACES.
*-----*
*--DEFAULT ECHO READQUEUE/QM
   05 WS-READ-QM-QUEUE.
      10 WS-QM-NAME PIC X(48) VALUE SPACES.
      10 WS-Q-NAME PIC X(48) VALUE
         'QUEUE1'.
*-----*
*--DEFAULT ECHO RESPONSE QUEUE/QM
   05 WS-RESPONSE-QM-QUEUE.
      10 WS-R-QM-NAME PIC X(48) VALUE SPACES.
      10 WS-R-Q-NAME PIC X(48) VALUE
         'IBM.REPLY.QUEUE'.
*-----*
*-----*
* ERROR MESSAGE FOR QUEUE
*-----*
01 WS-ERROR-MESSAGE.
   05 FILLER PIC X(5) VALUE
      'ECHO:'.
   05 FILLER PIC X(6) VALUE
      ' QID -'.
   05 WS-ERR-DISPLAY-QUEUE PIC X(30) VALUE SPACES.
   05 FILLER PIC X(6) VALUE
      ',CC -'.
   05 WS-ERR-DISPLAY-CCODE PIC 9(4) VALUE ZERO.
   05 FILLER PIC X(6) VALUE
      ',RC -'.
   05 WS-ERR-DISPLAY-RCODE PIC 9(4) VALUE ZERO.
   05 WS-FUNCTION PIC X(12) VALUE SPACES.

```

```

*
*-----*
*-----*
* ERROR WS VALUES
01 WS-ERR-INFO.
* COPY MQIERR.
*-----*
* - BEGIN - *** COPYBOOK: MQIERR *** - BEGIN - *
*-----*
* ERROR MODULE CALLING PARAMETERS
*-----*

02 ERR-HANDLER-COMMAREA.
05 ERR-CURRENT-INFO.
10 ERR-COM-HANDLER PIC X(48) VALUE SPACES.
10 ERR-QUEUE PIC X(48) VALUE SPACES.
10 ERR-FILE PIC X(8) VALUE SPACES.
10 ERR-DETAIL PIC X(80) VALUE SPACES.
10 ERR-DETAIL2 PIC X(80) VALUE SPACES.
10 ERR-Q-CODE PIC S9(8) COMP VALUE ZERO.
10 FILLER PIC X(8) VALUE SPACES.

05 ERR-RESULTS.
10 ERR-CODE PIC 9(6) VALUE ZERO.
10 FILLER PIC XX VALUE SPACES.
10 ERR-PROGRAM PIC X(8) VALUE SPACES.
10 ERR-TRANID PIC X(4) VALUE SPACES.
10 ERR-TERMID PIC X(4) VALUE SPACES.
10 ERR-TASKNO PIC S9(7) COMP-3 VALUE ZERO.
10 ERR-ABSTIME PIC S9(15) COMP-3 VALUE ZERO.

10 ERR-DEBUG-EIBFN PIC XX VALUE SPACES.
10 ERR-DEBUG-EIBRCODE PIC X(6) VALUE LOW-VALUES.
10 ERR-DEBUG-EIBRSRCE PIC X(8) VALUE LOW-VALUES.
10 ERR-DEBUG-EIBRESP PIC S9(8) COMP VALUE ZEROS.
10 ERR-DEBUG-EIBRESP2 PIC S9(8) COMP VALUE ZEROS.
10 ERR-DEBUG-EIBERRCD PIC X(4) VALUE LOW-VALUES.
10 ERR-DEBUG-ABEND PIC X(4) VALUE SPACES.
10 FILLER PIC X(12) VALUE SPACES.

*-----*
* - END - *** COPYBOOK: MQIERR *** - END - *
*-----*
* COPY MQIERRC.
*-----*
* IBM MQSERIES COMMON ERROR CODES
*-----*
01 MSG-ERROR-MESSAGES.
05 ERR-NO-ENVIRONMENT PIC 9(6) VALUE 900000.

05 ERR-CICS-ERROR PIC 9(6) VALUE 800000.
05 ERR-CICS-INVALID-REQ PIC 9(6) VALUE 800010.
05 ERR-CICS-ILLOGIC PIC 9(6) VALUE 800011.
05 ERR-CICS-ERROR-CHECKPOINT PIC 9(6) VALUE 800090.
05 ERR-CICS-ABEND PIC 9(6) VALUE 800099.
05 ERR-CICS-FILE-NOTOPEN PIC 9(6) VALUE 801012.
05 ERR-CICS-DISABLE PIC 9(6) VALUE 801019.
05 ERR-CICS-NO-STORAGE PIC 9(6) VALUE 802000.
05 ERR-CICS-LENGTH-ERR PIC 9(6) VALUE 803001.
05 ERR-CICS-MAPFAIL PIC 9(6) VALUE 808000.
05 ERR-CICS-PGMIDERR PIC 9(6) VALUE 809000.
05 ERR-CICS-FILEID PIC 9(6) VALUE 809010.
05 ERR-CICS-NOFILE PIC 9(6) VALUE 809011.
05 ERR-CICS-IO-ERROR PIC 9(6) VALUE 809012.
05 ERR-CICS-TRANIDERR PIC 9(6) VALUE 809050.

05 ERR-COM-FREE-ERROR PIC 9(6) VALUE 501001.
05 ERR-COM-EIB-ERROR PIC 9(6) VALUE 501002.
05 ERR-COM-STAT-ERROR PIC 9(6) VALUE 501003.
05 ERR-COM-ALLOC-ERROR PIC 9(6) VALUE 501004.
05 ERR-COM-ALLOC-RETRY PIC 9(6) VALUE 501005.
05 ERR-COM-CONN-ERROR PIC 9(6) VALUE 501006.
05 ERR-COM-SEND-ERROR PIC 9(6) VALUE 501008.
05 ERR-COM-RCV-RESP-ERR PIC 9(6) VALUE 501009.
05 ERR-COM-RESP-TYPE PIC 9(6) VALUE 501010.
05 ERR-COM-RESP-MSN PIC 9(6) VALUE 501011.
05 ERR-COM-RESP-FATAL PIC 9(6) VALUE 501012.
05 ERR-COM-MSG-ERROR PIC 9(6) VALUE 501013.
05 ERR-COM-BIG-INDIAN PIC 9(6) VALUE 501014.
05 ERR-COM-TSH-ERROR PIC 9(6) VALUE 501015.
05 ERR-COM-CCSID-ERROR PIC 9(6) VALUE 501016.
05 ERR-COM-MSH-ERROR PIC 9(6) VALUE 501017.

05 ERR-COM-MQX-ERROR PIC 9(6) VALUE 501018.
05 ERR-COM-INIT-ERROR PIC 9(6) VALUE 501019.
05 ERR-COM-FAP-ERROR PIC 9(6) VALUE 501020.
05 ERR-COM-MSG-SIZE PIC 9(6) VALUE 501021.
05 ERR-COM-WRAP-ERROR PIC 9(6) VALUE 501022.
05 ERR-COM-MCP-DOWN PIC 9(6) VALUE 501023.
05 ERR-COM-DOWN PIC 9(6) VALUE 501024.
05 ERR-COM-NOT-FOUND PIC 9(6) VALUE 501025.
05 ERR-COM-ERROR PIC 9(6) VALUE 501026.
05 ERR-COM-BUSY PIC 9(6) VALUE 501027.
05 ERR-COM-RESYNC-ERROR PIC 9(6) VALUE 501028.
05 ERR-COM-STATUS-ERROR PIC 9(6) VALUE 501029.
05 ERR-COM-LENGTH-ERROR PIC 9(6) VALUE 501030.
05 ERR-COM-MSG-PER-BATCH PIC 9(6) VALUE 501031.
05 ERR-COM-MAX-TRANSM-SIZE PIC 9(6) VALUE 501032.
05 ERR-COM-RESET-MSN PIC 9(6) VALUE 501050.

05 ERR-INT-LINK-ERROR PIC 9(6) VALUE 400000.
05 ERR-INT-LINK-COM-SIZE PIC 9(6) VALUE 400001.
05 ERR-INT-LINK-COM-DATA PIC 9(6) VALUE 400002.
05 ERR-INT-RETURN-ERROR PIC 9(6) VALUE 400003.
05 ERR-INT-MOVE-ERROR PIC 9(6) VALUE 400010.
05 ERR-INT-STRUC-MISSING PIC 9(6) VALUE 402000.
05 ERR-INT-STRUC-ERROR PIC 9(6) VALUE 402090.

05 ERR-LOGIC-NOT-SUPPORTED PIC 9(6) VALUE 300000.
05 ERR-LOGIC-STARTED-WRONG PIC 9(6) VALUE 300010.
05 ERR-LOGIC-REPEATED-FAILURE PIC 9(6) VALUE 300020.
05 ERR-LOGIC-LOCKS-EXCEEDED PIC 9(6) VALUE 300030.
05 ERR-LOGIC-MISSING-RECORD PIC 9(6) VALUE 301000.
05 ERR-LOGIC-RECORD-DUPLICATED PIC 9(6) VALUE 301010.
05 ERR-LOGIC-Q-CKP-MISSING PIC 9(6) VALUE 309010.

05 ERR-PROC-SYSTEM-STOPPED PIC 9(6) VALUE 100000.
05 ERR-PROC-SYSTEM-ACTIVE PIC 9(6) VALUE 100010.
05 ERR-PROC-SYS-START-NOQDR PIC 9(6) VALUE 100011.
05 ERR-PROC-SYS-START-MAXQDR PIC 9(6) VALUE 100012.
05 ERR-PROC-SYS-START-MAXCOM PIC 9(6) VALUE 100013.
05 ERR-PROC-SYS-START-NOSYS PIC 9(6) VALUE 100090.
05 ERR-PROC-Q-EXCEEDED-DEPTH PIC 9(6) VALUE 101000.
05 ERR-PROC-Q-CONCURRENT-UPD PIC 9(6) VALUE 101010.
05 ERR-PROC-Q-NOTFOUND PIC 9(6) VALUE 101015.
05 ERR-PROC-Q-STOPPED PIC 9(6) VALUE 101090.
05 ERR-PROC-Q-DISABLED PIC 9(6) VALUE 101091.
05 ERR-PROC-QSN-LIMIT-REACHED PIC 9(6) VALUE 102090.
05 ERR-PROC-FILE-SPACE-PUT PIC 9(6) VALUE 102091.
05 ERR-PROC-FILE-SPACE PIC 9(6) VALUE 102092.
05 ERR-PROC-DUAL-Q-ERROR PIC 9(6) VALUE 104021.
05 ERR-PROC-DUAL-Q-FILE PIC 9(6) VALUE 104022.
05 ERR-PROC-DUAL-Q-LOGIC PIC 9(6) VALUE 104023.
05 ERR-PROC-TRIGGER-ERROR PIC 9(6) VALUE 105090.
05 ERR-PROC-TRIGGER-DATA PIC 9(6) VALUE 105091.
05 ERR-PROC-NOT-AUTHORIZED PIC 9(6) VALUE 109000.

05 ERR-WARN-SYS-STARTED-W-ERR PIC 9(6) VALUE 010000.
05 ERR-WARN-SYS-STARTED-W-FILER PIC 9(6) VALUE 010001.
05 ERR-WARN-SYS-STARTED-W-COMER PIC 9(6) VALUE 010002.
05 ERR-WARN-SYS-STARTED-W-CHANG PIC 9(6) VALUE 010003.

05 ERR-WARN-COM-CONNECT PIC 9(6) VALUE 005000.
05 ERR-WARN-COM-OPENED PIC 9(6) VALUE 005001.
05 ERR-WARN-COM-QUEUE-OPENED PIC 9(6) VALUE 005002.
05 ERR-WARN-COM-LU62-CONNECT PIC 9(6) VALUE 005003.
05 ERR-WARN-COM-RECEIVER-ALLOC PIC 9(6) VALUE 005004.
05 ERR-WARN-COM-QUEUE-EMPTY PIC 9(6) VALUE 005005.
05 ERR-WARN-COM-QUEUE-CLOSED PIC 9(6) VALUE 005006.
05 ERR-WARN-COM-DISC PIC 9(6) VALUE 005007.
05 ERR-WARN-COM-SHUT PIC 9(6) VALUE 005008.
05 ERR-WARN-COM-SHUT-SENT PIC 9(6) VALUE 005009.

05 ERR-FUNCTION-STARTED PIC 9(6) VALUE 000100.
05 ERR-FUNCTION-DONE PIC 9(6) VALUE 001000.
05 ERR-FUNCTION-NOT-DONE PIC 9(6) VALUE 001090.

05 ERR-WARN-SYS-STARTED PIC 9(6) VALUE 000000.

05 SYNCH-MSN-ERROR PIC 9(6) VALUE 3.
05 SYNCH-MSG-DUP PIC 9(6) VALUE 4.
05 LU62-FREE-ERROR PIC 9(6) VALUE 10.
05 LU62-EIB-ERROR PIC 9(6) VALUE 11.
05 LU62-STAT-ERROR PIC 9(6) VALUE 12.
05 LU62-ALLOC-ERROR PIC 9(6) VALUE 13.

```

```

05 LU62-ALLOC-RETRY-ERROR      PIC 9(6)  VALUE 14.
05 LU62-CONN-ERROR             PIC 9(6)  VALUE 15.
05 LU62-SEND-ERROR             PIC 9(6)  VALUE 16.
05 LU62-RECV-RESP-ERROR       PIC 9(6)  VALUE 17.
05 INVLD-RESP-TYPE             PIC 9(6)  VALUE 23.
05 INVLD-RESP-MSN             PIC 9(6)  VALUE 24.
05 FATAL-RESP-TYPE            PIC 9(6)  VALUE 25.
05 RECOVERABLE-RESP-TYPE       PIC 9(6)  VALUE 26.
05 PARSE-MSN-ERROR             PIC 9(6)  VALUE 29.
05 PARSE-TYPE-ERROR            PIC 9(6)  VALUE 30.
05 PARSE-PDM-ERROR             PIC 9(6)  VALUE 31.
05 PARSE-SID-ERROR             PIC 9(6)  VALUE 32.
05 PARSE-PN-ERROR              PIC 9(6)  VALUE 33.
05 PARSE-KEY-ERROR             PIC 9(6)  VALUE 34.
05 PARSE-APIID-ERROR           PIC 9(6)  VALUE 35.
05 PARSE-ORG-DT-ERROR          PIC 9(6)  VALUE 38.
05 PARSE-ORIG-MSN-ERROR        PIC 9(6)  VALUE 39.
05 PARSE-BODY-ERROR            PIC 9(6)  VALUE 40.
05 PARSE-STATUS-ERROR          PIC 9(6)  VALUE 41.
05 PARSE-LENGTH-ERROR          PIC 9(6)  VALUE 42.
05 MCCONN-ERROR                PIC 9(6)  VALUE 51.
05 MQOPEN-ERROR                PIC 9(6)  VALUE 52.
05 MQGET-ERROR                  PIC 9(6)  VALUE 53.
05 MQPUT-ERROR                  PIC 9(6)  VALUE 54.
05 MQPT1-ERROR                  PIC 9(6)  VALUE 55.
05 MQCLOSE-ERROR                PIC 9(6)  VALUE 56.
05 MQDISC-ERROR                 PIC 9(6)  VALUE 57.
05 QM-OTHER-ERROR               PIC 9(6)  VALUE 60.
05 RECV-RETURN-LON-STATUS       PIC 9(6)  VALUE 80.
05 RECV-RETURN-LON-TYPE         PIC 9(6)  VALUE 81.
05 SIDRC-RETURN-MLP-FORMAT      PIC 9(6)  VALUE 91.

-----
*-----
* ENVIRONMENT
*-----
01 WS-ENVIR-INFO.
* COPY MQICENV.
*-----
* - BEGIN -      *** COPYBOOK: MQICENV ***      - BEGIN - *
*-----
* ENVIRONMENT VALUE      - SYSTEM (ENV)
*-----

02 ENV-DEFINITION.
03 ENV-DATA-FOR-SYSTEM.
05 ENV-PRODUCT-INSTALLED      PIC X(4) VALUE 'MQM '.
08 ENV-PRODUCT-MQM            VALUE 'MQM '.

05 ENV-PRODUCT-RUNTIME        PIC X(4) VALUE 'MQM '.
08 ENV-PRODUCT-RT-MQM         VALUE 'MQM '.

05 ENV-LANG-INFO.
10 ENV-LANGUAGE-FILE-CODE     PIC 99 VALUE 01.
10 ENV-LANGUAGE                PIC X(24)
                                VALUE 'ENGLISH'.

05 ENV-DATE-FORMAT            PIC 99 VALUE 01.
08 ENV-DATE-MMDDYY            VALUE 01.
08 ENV-DATE-YYMMDD            VALUE 02.
08 ENV-DATE-YYDDMM            VALUE 03.
08 ENV-DATE-YYDDD             VALUE 04.
08 ENV-DATE-DDMMYY            VALUE 05.

03 ENV-DATA-FOR-TRAN.

05 ENV-MASTER-TERMINAL-TRAN.
10 ENV-MT-MASTER-TASK-ID     PIC X(4) VALUE 'MQMT '.
10 ENV-MT-CONFIG-TASK-ID      PIC X(4) VALUE 'MQMC '.
10 ENV-MT-MONITOR-TASK-ID     PIC X(4) VALUE 'MQMM '.
10 ENV-MT-OPER-TASK-ID        PIC X(4) VALUE 'MQMO '.
10 ENV-MT-DISP-TASK-ID        PIC X(4) VALUE 'MQBQ '.
10 ENV-MT-QUEUE-TASK-ID       PIC X(4) VALUE 'MQMQ '.
10 ENV-MT-QUEUEI-TASK-ID      PIC X(4) VALUE 'MQDQ '.
10 ENV-MT-COM-TASK-ID         PIC X(4) VALUE 'MQMH '.
10 ENV-MT-COMI-TASK-ID        PIC X(4) VALUE 'MQDH '.
10 ENV-MT-SYS-TASK-ID         PIC X(4) VALUE 'MQMS '.
10 ENV-MT-SYSI-TASK-ID        PIC X(4) VALUE 'MQDS '.
10 ENV-MT-MONQ-TASK-ID        PIC X(4) VALUE 'MQQM '.
10 ENV-MT-MONC-TASK-ID        PIC X(4) VALUE 'MQCM '.
10 ENV-MT-SS-TASK-ID          PIC X(4) VALUE 'MQMA '.
10 ENV-MT-SC-TASK-ID          PIC X(4) VALUE 'MQMB '.

10 ENV-MT-SI-TASK-ID          PIC X(4) VALUE 'MQMI '.
10 ENV-MT-SR-TASK-ID          PIC X(4) VALUE 'MQMR '.
10 ENV-MT-SD-TASK-ID          PIC X(4) VALUE 'MQMD '.
10 FILLER                     PIC X(4) VALUE SPACES.
10 FILLER                     PIC X(4) VALUE SPACES.
10 FILLER                     PIC X(4) VALUE SPACES.

05 ENV-INTERNAL-ITEMS-TRAN.
10 ENV-II-MONITOR             PIC X(4) VALUE 'MQSM '.
10 ENV-II-M-RECOVERY          PIC X(4) VALUE 'MQSR '.
10 ENV-II-Q-RECOVERY          PIC X(4) VALUE 'MQSQ '.
10 ENV-II-START-STOP          PIC X(4) VALUE 'MQSS '.
10 ENV-II-TRAN-AIP2           PIC X(4) VALUE 'MQQ2 '.
10 ENV-II-TRAN-COM-CHECKP     PIC X(4) VALUE 'MQCP '.
10 ENV-II-TRAN-QUE-DELETE     PIC X(4) VALUE 'MQQD '.
10 ENV-II-TRAN-QUE-DEL-ALL    PIC X(4) VALUE 'MQQA '.
10 FILLER                     PIC X(4) VALUE SPACES.
10 FILLER                     PIC X(4) VALUE SPACES.
10 FILLER                     PIC X(4) VALUE SPACES.

03 ENV-DATA-FOR-PROGRAMS.

05 ENV-MASTER-TERMINAL-PROGRAMS.
10 ENV-MT-MASTER-PROGRAM      PIC X(8) VALUE 'MQPMP'.
10 ENV-MT-CONFIG-PROGRAM       PIC X(8) VALUE 'MQPMCFC'.
10 ENV-MT-MONITOR-PROGRAM      PIC X(8) VALUE 'MQPMMON'.
10 ENV-MT-OPER-PROGRAM         PIC X(8) VALUE 'MQPMOPR'.
10 ENV-MT-DISP-PROGRAM         PIC X(8) VALUE 'MQPDISP'.
10 ENV-MT-QUEUE-PROGRAM        PIC X(8) VALUE 'MQPMQUE'.
10 ENV-MT-QUEUEI-PROGRAM       PIC X(8) VALUE 'MQPMQUEI'.
10 ENV-MT-COM-PROGRAM          PIC X(8) VALUE 'MQPMCOM'.
10 ENV-MT-COMI-PROGRAM         PIC X(8) VALUE 'MQPMCOMI'.
10 ENV-MT-SYS-PROGRAM          PIC X(8) VALUE 'MQPMSYS'.
10 ENV-MT-SYSI-PROGRAM         PIC X(8) VALUE 'MQPMSYSI'.
10 ENV-MT-MONQ-PROGRAM         PIC X(8) VALUE 'MQPMMOQ'.
10 ENV-MT-MONC-PROGRAM         PIC X(8) VALUE 'MQPMMOC'.
10 ENV-MT-SS-PROGRAM           PIC X(8) VALUE 'MQPMSS'.
10 ENV-MT-SC-PROGRAM           PIC X(8) VALUE 'MQPMSC'.
10 ENV-MT-SI-PROGRAM           PIC X(8) VALUE 'MQPMSI'.
10 ENV-MT-SR-PROGRAM           PIC X(8) VALUE 'MQPMSM'.
10 ENV-MT-SD-PROGRAM           PIC X(8) VALUE 'MQPMDL'.
10 ENV-MT-CMD-PROGRAM          PIC X(8) VALUE 'MQPCMD'.
10 FILLER                     PIC X(8) VALUE SPACES.
10 FILLER                     PIC X(8) VALUE SPACES.

05 ENV-INTERNAL-ITEMS-PROGRAMS.
10 ENV-II-LINK-ERROR           PIC X(8) VALUE 'MQPERR '.
10 ENV-II-LINK-EIB1           PIC X(8) VALUE 'MQPEIB1 '.
10 ENV-II-LINK-AIP0           PIC X(8) VALUE 'MQPAIP0 '.
10 ENV-II-LINK-AIP1           PIC X(8) VALUE 'MQPAIP1 '.
10 ENV-II-LINK-AIP2           PIC X(8) VALUE 'MQPAIP2 '.

10 ENV-II-LINK-ECHO            PIC X(8) VALUE 'MQPECHO '.
10 ENV-II-LINK-FINDQ           PIC X(8) VALUE 'MQPFINDQ '.
10 ENV-II-LINK-QUE1            PIC X(8) VALUE 'MQPQUE1 '.
10 ENV-II-LINK-QUE2            PIC X(8) VALUE 'MQPQUE2 '.
10 ENV-II-LINK-INIT1           PIC X(8) VALUE 'MQPINIT1 '.
10 ENV-II-LINK-INIT2           PIC X(8) VALUE 'MQPINIT2 '.
10 ENV-II-LINK-SSQ             PIC X(8) VALUE 'MQPSSQ '.
10 ENV-II-LINK-SCHK            PIC X(8) VALUE 'MQPSCHK '.
10 ENV-II-LINK-SREC            PIC X(8) VALUE 'MQPSREC '.
10 ENV-II-LINK-QRECOVERY       PIC X(8) VALUE 'MQPQREC '.
10 ENV-II-LINK-SENDER          PIC X(8) VALUE 'MQPSEND '.
10 ENV-II-LINK-RECIEVER        PIC X(8) VALUE 'MQPRECV '.
10 ENV-II-LINK-COM-CHECKP     PIC X(8) VALUE 'MQPCCKPT'.
10 ENV-II-LINK-QUE-DELETE     PIC X(8) VALUE 'MQPQDEL '.
10 ENV-II-LINK-SET-MAP         PIC X(8) VALUE 'MQPSMAP '.
10 ENV-II-LINK-LU21            PIC X(8) VALUE 'MQPLU21 '.
10 ENV-II-LINK-LU33            PIC X(8) VALUE 'MQPLU33 '.
10 FILLER                     PIC X(8) VALUE SPACES.
10 FILLER                     PIC X(8) VALUE SPACES.
10 FILLER                     PIC X(8) VALUE SPACES.

03 ENV-DATA-FOR-MAPS.

05 ENV-MASTER-TERMINAL-MAPS.
10 ENV-MT-MASTER-MAPSCREEN    PIC X(8) VALUE 'MQMMP'.
10 ENV-MT-CONFIG-MAPSCREEN     PIC X(8) VALUE 'MQMCMFC'.
10 ENV-MT-MONITOR-MAPSCREEN    PIC X(8) VALUE 'MQMMMON'.
10 ENV-MT-OPER-MAPSCREEN       PIC X(8) VALUE 'MQMMOPR'.
10 ENV-MT-DISP-MAPSCREEN       PIC X(8) VALUE 'MQMDISP'.

```

```

10 ENV-MT-QUEUE-MAPSCREEN PIC X(8) VALUE 'MQMMQUE'.
10 ENV-MT-QUEUEI-MAPSCREEN PIC X(8) VALUE 'MQMMQUE'.
10 ENV-MT-COM-MAPSCREEN PIC X(8) VALUE 'MQMMCOM'.
10 ENV-MT-COMI-MAPSCREEN PIC X(8) VALUE 'MQMMCOM'.
10 ENV-MT-SYS-MAPSCREEN PIC X(8) VALUE 'MQMMSYS'.
10 ENV-MT-SYSI-MAPSCREEN PIC X(8) VALUE 'MQMMSYS'.
10 ENV-MT-MOQ-MAPSCREEN PIC X(8) VALUE 'MQMMMOQ'.
10 ENV-MT-MONC-MAPSCREEN PIC X(8) VALUE 'MQMMMOC'.
10 ENV-MT-SS-MAPSCREEN PIC X(8) VALUE 'MQMMSS'.
10 ENV-MT-SC-MAPSCREEN PIC X(8) VALUE 'MQMMSC'.
10 ENV-MT-SI-MAPSCREEN PIC X(8) VALUE 'MQMMSI'.
10 ENV-MT-SR-MAPSCREEN PIC X(8) VALUE 'MQMMMSN'.
10 ENV-MT-SD-MAPSCREEN PIC X(8) VALUE 'MQMMDEL'.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.
10 FILLER PIC X(8) VALUE SPACES.

03 ENV-DATA-FOR-CONSTANTS.

05 ENV-CONFIG-DDNAME PIC X(8) VALUE 'MQFCNFG'.
05 ENV-SYSTEM-NUMBER PIC 9(4) VALUE 1.
05 ENV-MASTER-TERMINAL-CONS.
  10 ENV-MT-TITLE PIC X(40) VALUE
  ' IBM MQSeries for VSE/ESA Version 2 '.

05 ENV-INTERNAL-ITEMS-CONS.
  10 ENV-II-ERROR-TD PIC X(4) VALUE 'MQR'.
  10 ENV-II-ERROR-CSMT PIC X(4) VALUE 'CSMT'.
  10 ENV-II-SYSTEM-ANCHOR PIC X(8) VALUE 'MQTAQM'.
  10 ENV-II-SYSTEM-PREFIX PIC X(4) VALUE 'MQI '.
  10 ENV-II-DUMPCODE PIC X(4) VALUE 'MQ??'.
  10 ENV-II-ENQ-INIT1 PIC X(8) VALUE 'MQPINIT1'.
  10 ENV-II-SYSTEM-ENVIR PIC X(8) VALUE 'MQTENV '.
  10 ENV-IT-UN-INIT-MSG PIC X(80) VALUE
  'MQ900000: MQSERIES VSE ENVIRONMENT not initialized.'.
  10 FILLER PIC X(80) VALUE SPACES.

-----*
* - END - *** COPYBOOK: MQICENV *** - END - *
-----*

-----*
* USER PROCESS DEFINITON
-----*
01 WS-PROC.
* COPY CMQTMV.
*****
**
** FILE NAME: CMQTMV **
**
** DESCRIPTIVE NAME: COBOL copy file for MQTM structure **
**
** VERSION 1.3.0 **
**
** FUNCTION: This file declares the MQTM structure, **
** which forms part of the IBM Message **
** Queue Interface (MQI). **
*****

** MQTM structure
  10 MQTM.
** Structure identifier
  15 MQTM-STRUCID PIC X(4) VALUE 'TM '.
** Structure version number
  15 MQTM-VERSION PIC S9(9) BINARY VALUE 1.
** Name of triggered queue
  15 MQTM-QNAME.
  25 MQI-PROC-LOCAL-QUEUE-NAME PIC X(48) VALUE SPACE.

** Name of process object
  15 MQTM-PROCESSNAME PIC X(48) VALUE SPACES.

** Trigger data
  15 MQTM-TRIGGERDATA PIC X(64) VALUE SPACES.
  15 MQTM-TRIGGERDATA-RED REDEFINES MQTM-TRIGGERDATA.
  25 MQI-PROC-TRANS-ID PIC X(4).
  25 MQI-PROC-PROGRAM-ID PIC X(8).

  25 MQI-PROC-TRIGGER-EVENT PIC X.
  88 MQI-PROC-TRIGGER-FIRST VALUE 'F'.
  88 MQI-PROC-TRIGGER-EVERY VALUE 'E'.

** Application type
  15 MQTM-APPLTYPE PIC S9(9) BINARY VALUE 0.
** Application identifier
  15 MQTM-APPLID PIC X(256) VALUE SPACES.
** Environment data
  15 MQTM-ENVDATA PIC X(128) VALUE SPACES.
** User data
  15 MQTM-USERDATA PIC X(128) VALUE SPACES.
  15 MQTM-USERDATA-RED REDEFINES MQTM-USERDATA.
  25 MQI-PROC-CHANNEL-NAME PIC X(20).

-----*
-----*
01 MQI-VALUES.
-----*
* COPY CMQV.
*****
**
** FILE NAME: CMQV **
**
** DESCRIPTIVE NAME: COBOL copy file for MQI constants **
**
** VERSION 1.3.0 **
**
** FUNCTION: This file declares the constants **
** which form part of the IBM Message **
** Queue Interface (MQI). **
*****

** Values Related to MQDLH Structure **
** Structure Identifier **
  10 MQDLH-STRUC-ID PIC X(4) VALUE 'DLH '.

** Structure Version Number
  10 MQDLH-VERSION-1 PIC S9(9) BINARY VALUE 1.

*****

** Values Related to MQGMO Structure **
*****

** Structure Identifier
  10 MQGMO-STRUC-ID PIC X(4) VALUE 'GMO '.

** Structure Version Number
  10 MQGMO-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Get-Message Options
  10 MQGMO-WAIT PIC S9(9) BINARY VALUE 1.
  10 MQGMO-NO-WAIT PIC S9(9) BINARY VALUE 0.
  10 MQGMO-BROWSE-FIRST PIC S9(9) BINARY VALUE 16.
  10 MQGMO-BROWSE-NEXT PIC S9(9) BINARY VALUE 32.
  10 MQGMO-ACCEPT-TRUNCATED-MSG PIC S9(9) BINARY VALUE 64.
  10 MQGMO-SET-SIGNAL PIC S9(9) BINARY VALUE 8.
  10 MQGMO-SYNCPPOINT PIC S9(9) BINARY VALUE 2.
  10 MQGMO-NO-SYNCPPOINT PIC S9(9) BINARY VALUE 4.
  10 MQGMO-MSG-UNDER-CURSOR PIC S9(9) BINARY VALUE 256.
  10 MQGMO-LOCK PIC S9(9) BINARY VALUE 512.
  10 MQGMO-UNLOCK PIC S9(9) BINARY VALUE 1024.

** Wait Interval
  10 MQWI-UNLIMITED PIC S9(9) BINARY VALUE -1.

*****

** Values Related to MQMD Structure **
*****

** Structure Identifier
  10 MQMD-STRUC-ID PIC X(4) VALUE 'MD '.

** Structure Version Number
  10 MQMD-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Report Options
  10 MQRO-NONE PIC S9(9) BINARY VALUE 0.

```

```

** Message Types
10 MQMT-REQUEST PIC S9(9) BINARY VALUE 1.
10 MQMT-REPLY PIC S9(9) BINARY VALUE 2.
10 MQMT-DATAGRAM PIC S9(9) BINARY VALUE 8.
10 MQMT-REPORT PIC S9(9) BINARY VALUE 4.

** Expiry Value
10 MQEI-UNLIMITED PIC S9(9) BINARY VALUE -1.

** Feedback Values
10 MQFB-NONE PIC S9(9) BINARY VALUE 0.
10 MQFB-QUIT PIC S9(9) BINARY VALUE 256.
10 MQFB-SYSTEM-FIRST PIC S9(9) BINARY VALUE 1.
10 MQFB-SYSTEM-LAST PIC S9(9) BINARY VALUE 65535.
10 MQFB-APPL-FIRST PIC S9(9) BINARY VALUE 65536.
10 MQFB-APPL-LAST PIC S9(9) BINARY VALUE 999999999.

* format
10 MQFMT-NONE PIC X(8) VALUE SPACES.
10 MQFMT-DEAD-LETTER-Q-HEADER PIC X(8) VALUE 'MQDLQH'.
10 MQFMT-TRIGGER PIC X(8) VALUE 'MQTRIG'.
10 MQFMT-XMIT-Q-HEADER PIC X(8) VALUE 'MQXMIT'.

** Encoding Value
10 MQENC-NATIVE PIC S9(9) BINARY VALUE 785.

** Encoding Masks
10 MQENC-INTEGGER-MASK PIC S9(9) BINARY VALUE 15.
10 MQENC-DECIMAL-MASK PIC S9(9) BINARY VALUE 240.
10 MQENC-FLOAT-MASK PIC S9(9) BINARY VALUE 3840.
10 MQENC-RESERVED-MASK PIC S9(9) BINARY VALUE -4096.

** Encodings for Binary Integers
10 MQENC-INTEGGER-UNDEFINED PIC S9(9) BINARY VALUE 0.
10 MQENC-INTEGGER-NORMAL PIC S9(9) BINARY VALUE 1.
10 MQENC-INTEGGER-REVERSED PIC S9(9) BINARY VALUE 2.

** Encodings for Packed-Decimal Integers
10 MQENC-DECIMAL-UNDEFINED PIC S9(9) BINARY VALUE 0.
10 MQENC-DECIMAL-NORMAL PIC S9(9) BINARY VALUE 16.
10 MQENC-DECIMAL-REVERSED PIC S9(9) BINARY VALUE 32.

** Encodings for Floating-Point Numbers
10 MQENC-FLOAT-UNDEFINED PIC S9(9) BINARY VALUE 0.
10 MQENC-FLOAT-IEEE-NORMAL PIC S9(9) BINARY VALUE 256.
10 MQENC-FLOAT-IEEE-REVERSED PIC S9(9) BINARY VALUE 512.
10 MQENC-FLOAT-S390 PIC S9(9) BINARY VALUE 768.

** Coded Character-Set Identifier
10 MQCCSI-Q-MGR PIC S9(9) BINARY VALUE 0.

** Persistence Values
10 MQPER-PERSISTENT PIC S9(9) BINARY VALUE 1.
10 MQPER-PERSISTENCE-AS-Q-DEF PIC S9(9) BINARY VALUE 2.

** Message Id Value
10 MQMI-NONE PIC X(24) VALUE LOW-VALUES.

** Correllation Id Value
10 MQCI-NONE PIC X(24) VALUE LOW-VALUES.

*****
** Values Related to MQOD Structure **
*****

** Structure Identifier
10 MQOD-STRUC-ID PIC X(4) VALUE 'OD '.

** Structure Version Number
10 MQOD-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Object Types
10 MQOT-Q PIC S9(9) BINARY VALUE 1.

*****
** Values Related to MQPMO Structure **
*****

** Structure Identifier
10 MQPMO-STRUC-ID PIC X(4) VALUE 'PMO '.

** Structure Version Number
10 MQPMO-VERSION-1 PIC S9(9) BINARY VALUE 1.

** Put-Message Options
10 MQPMO-SYNCPPOINT PIC S9(9) BINARY VALUE 2.
10 MQPMO-NO-SYNCPPOINT PIC S9(9) BINARY VALUE 4.

*****
** Values Related to MQTM Structure **
*****

** Structure Identifier
10 MQTM-STRUC-ID PIC X(4) VALUE 'TM '.

** Structure Version Number
10 MQTM-VERSION-1 PIC S9(9) BINARY VALUE 1.

*****
** Values Related to MQCLOSE Call **
*****

** Close Options
10 MQCO-NONE PIC S9(9) BINARY VALUE 0.

*****
** Values Related to MQINQ Call **
*****

** Character-Attribute Selectors
10 MQCA-BASE-Q-NAME PIC S9(9) BINARY VALUE 2002.
10 MQCA-CREATION-DATE PIC S9(9) BINARY VALUE 2004.
10 MQCA-CREATION-TIME PIC S9(9) BINARY VALUE 2005.
10 MQCA-FIRST PIC S9(9) BINARY VALUE 2001.
10 MQCA-INITIATION-Q-NAME PIC S9(9) BINARY VALUE 2008.
10 MQCA-LAST PIC S9(9) BINARY VALUE 4000.
10 MQCA-PROCESS-NAME PIC S9(9) BINARY VALUE 2012.
10 MQCA-Q-DESC PIC S9(9) BINARY VALUE 2013.
10 MQCA-Q-NAME PIC S9(9) BINARY VALUE 2016.
10 MQCA-REMOTE-Q-MGR-NAME PIC S9(9) BINARY VALUE 2017.
10 MQCA-REMOTE-Q-NAME PIC S9(9) BINARY VALUE 2018.

** Integer-Attribute Selectors
10 MQIA-CURRENT-Q-DEPTH PIC S9(9) BINARY VALUE 3.
10 MQIA-DEF-PERSISTENCE PIC S9(9) BINARY VALUE 5.
10 MQIA-DEFINITION-TYPE PIC S9(9) BINARY VALUE 7.
10 MQIA-FIRST PIC S9(9) BINARY VALUE 1.
10 MQIA-INHIBIT-GET PIC S9(9) BINARY VALUE 9.
10 MQIA-INHIBIT-PUT PIC S9(9) BINARY VALUE 10.
10 MQIA-LAST PIC S9(9) BINARY VALUE 2000.
10 MQIA-MAX-MSG-LENGTH PIC S9(9) BINARY VALUE 13.
10 MQIA-MAX-Q-DEPTH PIC S9(9) BINARY VALUE 15.
10 MQIA-OPEN-INPUT-COUNT PIC S9(9) BINARY VALUE 17.
10 MQIA-OPEN-OUTPUT-COUNT PIC S9(9) BINARY VALUE 18.
10 MQIA-Q-TYPE PIC S9(9) BINARY VALUE 20.
10 MQIA-SHAREABILITY PIC S9(9) BINARY VALUE 23.
10 MQIA-TRIGGER-CONTROL PIC S9(9) BINARY VALUE 24.
10 MQIA-TRIGGER-TYPE PIC S9(9) BINARY VALUE 28.
10 MQIA-USAGE PIC S9(9) BINARY VALUE 12.

** Integer Attribute Value Denoting 'Not Applicable'
10 MQIAV-NOT-APPLICABLE PIC S9(9) BINARY VALUE -1.

*****
** Values Related to MQOPEN Call **
*****

** Open Options
10 MQOO-INPUT-SHARED PIC S9(9) BINARY VALUE 2.
10 MQOO-INPUT-EXCLUSIVE PIC S9(9) BINARY VALUE 4.
10 MQOO-BROWSE PIC S9(9) BINARY VALUE 8.
10 MQOO-OUTPUT PIC S9(9) BINARY VALUE 16.
10 MQOO-INQUIRE PIC S9(9) BINARY VALUE 32.

*****
** Values Related to All Calls **
*****

** String Lengths

```

10 MQ-CREATION-DATE-LENGTH	PIC S9(9) BINARY	VALUE 12.	10 MQRC-XMIT-Q-TYPE-ERROR	PIC S9(9) BINARY	VALUE 2091.
10 MQ-CREATION-TIME-LENGTH	PIC S9(9) BINARY	VALUE 8.	10 MQRC-XMIT-Q-USAGE-ERROR	PIC S9(9) BINARY	VALUE 2092.
10 MQ-PROCESS-APPL-ID-LENGTH	PIC S9(9) BINARY	VALUE 256.	10 MQRC-PMO-ERROR	PIC S9(9) BINARY	VALUE 2173.
10 MQ-PROCESS-DESC-LENGTH	PIC S9(9) BINARY	VALUE 64.	10 MQRC-GMO-ERROR	PIC S9(9) BINARY	VALUE 2186.
10 MQ-PROCESS-ENV-DATA-LENGTH	PIC S9(9) BINARY	VALUE 128.			
10 MQ-PROCESS-NAME-LENGTH	PIC S9(9) BINARY	VALUE 48.	10 MQRC-UNEXPECTED-ERROR	PIC S9(9) BINARY	VALUE 2195.
10 MQ-PROCESS-USER-DATA-LENGTH	PIC S9(9) BINARY	VALUE 128.	10 MQRC-MSG-ID-ERROR	PIC S9(9) BINARY	VALUE 2206.
10 MQ-Q-DESC-LENGTH	PIC S9(9) BINARY	VALUE 64.	10 MQRC-CORREL-ID-ERROR	PIC S9(9) BINARY	VALUE 2207.
10 MQ-Q-NAME-LENGTH	PIC S9(9) BINARY	VALUE 48.			
10 MQ-Q-MGR-DESC-LENGTH	PIC S9(9) BINARY	VALUE 64.	10 MQRC-FILE-SYSTEM-ERROR	PIC S9(9) BINARY	VALUE 2208.
10 MQ-Q-MGR-NAME-LENGTH	PIC S9(9) BINARY	VALUE 48.	10 MQRC-NO-MSG-LOCKED	PIC S9(9) BINARY	VALUE 2209.
10 MQ-TRIGGER-DATA-LENGTH	PIC S9(9) BINARY	VALUE 64.			

** Completion Codes

10 MQCC-OK	PIC S9(9) BINARY	VALUE 0.
10 MQCC-WARNING	PIC S9(9) BINARY	VALUE 1.
10 MQCC-FAILED	PIC S9(9) BINARY	VALUE 2.

** Reason Codes

10 MQRC-NONE	PIC S9(9) BINARY	VALUE 0.
10 MQRC-ACCESS-RESTRICTED	PIC S9(9) BINARY	VALUE 2000.
10 MQRC-ALIAS-BASE-Q-TYPE-ERROR	PIC S9(9) BINARY	VALUE 2001.
10 MQRC-ALREADY-CONNECTED	PIC S9(9) BINARY	VALUE 2002.
10 MQRC-BUFFER-ERROR	PIC S9(9) BINARY	VALUE 2004.
10 MQRC-BUFFER-LENGTH-ERROR	PIC S9(9) BINARY	VALUE 2005.
10 MQRC-CHAR-ATTR-LENGTH-ERROR	PIC S9(9) BINARY	VALUE 2006.
10 MQRC-CHAR-ATTRS-ERROR	PIC S9(9) BINARY	VALUE 2007.
10 MQRC-CHAR-ATTRS-TOO-SHORT	PIC S9(9) BINARY	VALUE 2008.
10 MQRC-CONNECTION-BROKEN	PIC S9(9) BINARY	VALUE 2009.
10 MQRC-DATA-LENGTH-ERROR	PIC S9(9) BINARY	VALUE 2010.
10 MQRC-EXPIRY-ERROR	PIC S9(9) BINARY	VALUE 2013.
10 MQRC-FEEDBACK-ERROR	PIC S9(9) BINARY	VALUE 2014.
10 MQRC-GET-INHIBITED	PIC S9(9) BINARY	VALUE 2016.
10 MQRC-HANDLE-NOT-AVAILABLE	PIC S9(9) BINARY	VALUE 2017.
10 MQRC-HCONN-ERROR	PIC S9(9) BINARY	VALUE 2018.
10 MQRC-HOBJ-ERROR	PIC S9(9) BINARY	VALUE 2019.
10 MQRC-INT-ATTR-COUNT-ERROR	PIC S9(9) BINARY	VALUE 2021.
10 MQRC-INT-ATTR-COUNT-TOO-SMALL	PIC S9(9) BINARY	VALUE 2022.
10 MQRC-INT-ATTRS-ARRAY-ERROR	PIC S9(9) BINARY	VALUE 2023.
10 MQRC-MAX-CONNS-LIMIT-REACHED	PIC S9(9) BINARY	VALUE 2025.
10 MQRC-MD-ERROR	PIC S9(9) BINARY	VALUE 2026.
10 MQRC-MISSING-REPLY-TO-Q	PIC S9(9) BINARY	VALUE 2027.
10 MQRC-MSG-TYPE-ERROR	PIC S9(9) BINARY	VALUE 2029.
10 MQRC-MSG-TOO-BIG-FOR-Q	PIC S9(9) BINARY	VALUE 2030.
10 MQRC-NO-MSG-AVAILABLE	PIC S9(9) BINARY	VALUE 2033.
10 MQRC-NO-MSG-UNDER-CURSOR	PIC S9(9) BINARY	VALUE 2034.
10 MQRC-NOT-AUTHORIZED	PIC S9(9) BINARY	VALUE 2035.
10 MQRC-NOT-OPEN-FOR-BROWSE	PIC S9(9) BINARY	VALUE 2036.
10 MQRC-NOT-OPEN-FOR-INPUT	PIC S9(9) BINARY	VALUE 2037.
10 MQRC-NOT-OPEN-FOR-INQUIRE	PIC S9(9) BINARY	VALUE 2038.
10 MQRC-NOT-OPEN-FOR-OUTPUT	PIC S9(9) BINARY	VALUE 2039.
10 MQRC-OBJECT-CHANGED	PIC S9(9) BINARY	VALUE 2041.
10 MQRC-OBJECT-IN-USE	PIC S9(9) BINARY	VALUE 2042.
10 MQRC-OBJECT-TYPE-ERROR	PIC S9(9) BINARY	VALUE 2043.
10 MQRC-OD-ERROR	PIC S9(9) BINARY	VALUE 2044.
10 MQRC-OPTION-NOT-VALID-FOR-TYPE	PIC S9(9) BINARY	VALUE 2045.
10 MQRC-OPTIONS-ERROR	PIC S9(9) BINARY	VALUE 2046.
10 MQRC-PERSISTENCE-ERROR	PIC S9(9) BINARY	VALUE 2047.
10 MQRC-PRIORITY-EXCEEDS-MAXIMUM	PIC S9(9) BINARY	VALUE 2049.
10 MQRC-PRIORITY-ERROR	PIC S9(9) BINARY	VALUE 2050.
10 MQRC-PUT-INHIBITED	PIC S9(9) BINARY	VALUE 2051.
10 MQRC-Q-FULL	PIC S9(9) BINARY	VALUE 2053.
10 MQRC-Q-SPACE-NOT-AVAILABLE	PIC S9(9) BINARY	VALUE 2056.
10 MQRC-Q-MGR-NAME-ERROR	PIC S9(9) BINARY	VALUE 2058.
10 MQRC-Q-MGR-NOT-AVAILABLE	PIC S9(9) BINARY	VALUE 2059.
10 MQRC-REPORT-OPTIONS-ERROR	PIC S9(9) BINARY	VALUE 2061.
10 MQRC-SECURITY-ERROR	PIC S9(9) BINARY	VALUE 2063.
10 MQRC-SELECTOR-COUNT-ERROR	PIC S9(9) BINARY	VALUE 2065.
10 MQRC-SELECTOR-LIMIT-EXCEEDED	PIC S9(9) BINARY	VALUE 2066.
10 MQRC-SELECTOR-ERROR	PIC S9(9) BINARY	VALUE 2067.
10 MQRC-SELECTOR-NOT-FOR-TYPE	PIC S9(9) BINARY	VALUE 2068.
10 MQRC-SIGNAL-OUTSTANDING	PIC S9(9) BINARY	VALUE 2069.
10 MQRC-SIGNAL-REQUEST-ACCEPTED	PIC S9(9) BINARY	VALUE 2070.
10 MQRC-STORAGE-NOT-AVAILABLE	PIC S9(9) BINARY	VALUE 2071.
10 MQRC-SYNCPPOINT-NOT-AVAILABLE	PIC S9(9) BINARY	VALUE 2072.
10 MQRC-TRUNCATED-MSG-ACCEPTED	PIC S9(9) BINARY	VALUE 2079.
10 MQRC-TRUNCATED-MSG-FAILED	PIC S9(9) BINARY	VALUE 2080.
10 MQRC-UNEXPECTED-CONNECT-ERROR	PIC S9(9) BINARY	VALUE 2081.
10 MQRC-UNKNOWN-ALIAS-BASE-Q	PIC S9(9) BINARY	VALUE 2082.
10 MQRC-UNKNOWN-OBJECT-NAME	PIC S9(9) BINARY	VALUE 2085.
10 MQRC-UNKNOWN-OBJECT-Q-MGR	PIC S9(9) BINARY	VALUE 2086.
10 MQRC-UNKNOWN-REMOTE-Q-MGR	PIC S9(9) BINARY	VALUE 2087.
10 MQRC-WAIT-INTERVAL-ERROR	PIC S9(9) BINARY	VALUE 2090.

 ** Values Related to Queue Attributes **

** Queue Types

10 MQQT-LOCAL	PIC S9(9) BINARY	VALUE 1.
10 MQQT-ALIAS	PIC S9(9) BINARY	VALUE 3.
10 MQQT-REMOTE	PIC S9(9) BINARY	VALUE 6.

** Queue Definition Types

10 MQQDT-PREDEFINED	PIC S9(9) BINARY	VALUE 1.
---------------------	------------------	----------

** Inhibit Get

10 MQQA-GET-INHIBITED	PIC S9(9) BINARY	VALUE 1.
10 MQQA-GET-ALLOWED	PIC S9(9) BINARY	VALUE 0.

** Inhibit Put

10 MQQA-PUT-INHIBITED	PIC S9(9) BINARY	VALUE 1.
10 MQQA-PUT-ALLOWED	PIC S9(9) BINARY	VALUE 0.

** Queue Shareability

10 MQQA-SHAREABLE	PIC S9(9) BINARY	VALUE 1.
10 MQQA-NOT-SHAREABLE	PIC S9(9) BINARY	VALUE 0.

** Message Delivery Sequence

10 MQMDS-FIFO	PIC S9(9) BINARY	VALUE 1.
---------------	------------------	----------

** Trigger Control

10 MQTC-OFF	PIC S9(9) BINARY	VALUE 0.
10 MQTC-ON	PIC S9(9) BINARY	VALUE 1.

** Trigger Types

10 MQTT-NONE	PIC S9(9) BINARY	VALUE 0.
10 MQTT-FIRST	PIC S9(9) BINARY	VALUE 1.
10 MQTT-EVERY	PIC S9(9) BINARY	VALUE 2.

** Queue Usage

10 MQUS-NORMAL	PIC S9(9) BINARY	VALUE 0.
10 MQUS-TRANSMISSION	PIC S9(9) BINARY	VALUE 1.

 ** Values Related to Process-Definition Attributes **

** Application Type

10 MQAT-USER-FIRST	PIC S9(9) BINARY	VALUE 65536.
10 MQAT-USER-LAST	PIC S9(9) BINARY	VALUE 999999999.

*

10 MQAT-OS2	PIC S9(9) BINARY	VALUE 4.
10 MQAT-DOS	PIC S9(9) BINARY	VALUE 5.
10 MQAT-AIX	PIC S9(9) BINARY	VALUE 6.
10 MQAT-OS400	PIC S9(9) BINARY	VALUE 8.
10 MQAT-WINDOWS	PIC S9(9) BINARY	VALUE 9.
10 MQAT-CICS-VSE	PIC S9(9) BINARY	VALUE 10.
10 MQAT-VMS	PIC S9(9) BINARY	VALUE 12.
10 MQAT-GUARDIAN	PIC S9(9) BINARY	VALUE 13.
10 MQAT-VOS	PIC S9(9) BINARY	VALUE 14.

 ** Values Related to Queue-Manager Attributes **

** Syncpoint Availability

10 MQSP-AVAILABLE	PIC S9(9) BINARY	VALUE 1.
-------------------	------------------	----------

```

-----*
* API
* COPY MQIAIP1.
* - BEGIN - *** COPYBOOK: MQIAIP1 *** - BEGIN - *
* 9/ 1/93 REV:
* APPL. INTERFACE PARM FOR SSI STUBS
-----*

05 API-CALL-PARM.
10 API-FUNCTION PIC X(4).
88 API-CONNECT VALUE 'CONN', 'CONI'
'MCCO'.
88 API-CONNECT-VIA-APPL VALUE 'CONN', 'CONI'.
88 API-CONNECT-VIA-INTERFACE VALUE 'CONI'.
88 API-MCP-CONNECT VALUE 'MCCO'.
88 API-OPEN VALUE 'OPEN'.
88 API-PUT VALUE 'PUT '.
88 API-INQ VALUE 'INQ '.
88 API-GET VALUE 'GET '.
88 API-GET-QSN VALUE 'GETQ'.
88 API-CLOSE VALUE 'CLOS'.
88 API-DISCONNECT VALUE 'DISC'.

10 API-RETURN-CODE-INFO.
15 API-CCODE-ADDR USAGE POINTER.
15 API-RCODE-ADDR USAGE POINTER.

10 API-VARIABLE-PARM-INFO.
15 API-HCONN-ADDR USAGE POINTER.
15 API-HOBJ-ADDR USAGE POINTER.
15 API-PARM-NUM PIC S9(4) COMP.
15 FILLER PIC XX.
15 API-PARM-ADDR-LIST.
20 API-PARM-ADDR OCCURS 50 TIMES
USAGE POINTER.

-----*
* - END - *** COPYBOOK: MQIAIP1 *** - END - *
-----*
* COPY MQIENQ.
* "MQIENQ"
* ENQ/DEQ DEFINITIONS FOR QUEUEING/COM. HANDLERS *
-----*
01 ENQ-INFO.

---GLOBAL ENVIRONMENT TS QUEUE ID
05 ENQ-ENVIR-TS-INFO.
10 ENQ-ENVIR-TS-ITEM PIC S9(4) COMP VALUE +1.
10 ENQ-ENVIR-TS-SIZE PIC S9(4) COMP VALUE ZERO.
10 ENQ-ENVIR-TS-QID PIC X(8) VALUE 'MQSERIES'.

---ENQ KEY FOR LOCKING
05 ENQ-RECORD.
10 ENQ-QSN PIC 9(10) VALUE ZERO.
10 ENQ-OBJ-NAME PIC X(48) VALUE SPACES.

05 QSN-BUSY-FLAG PIC X VALUE SPACE.
88 QSN-BUSY VALUE 'Y'.
88 QSN-BUSY-OK VALUE 'N'.

---QUE RECORD RIB KEY
05 QUEUE-KEY.
10 QUEUE-KEY-OBJ PIC X(48) VALUE SPACES.
10 QUEUE-KEY-QSN PIC S9(8) COMP VALUE ZERO.

---DRQ TS QUEUE ID
05 ENQ-RT-QUEUE-ID.
10 ENQ-RT-CONSTANT PIC X(3) VALUE 'MQT'.
10 ENQ-RT-TYPE PIC X VALUE '0'.
10 ENQ-RT-HHHH PIC 9999 VALUE ZERO.
10 ENQ-RT-ITEM PIC 9999 VALUE ZERO.

---DRQ WAIT REQID
05 ENQ-RT-REQID-ID.
10 ENQ-RT-R-CONSTANT PIC X(3) VALUE 'MQT'.
10 ENQ-RT-R-TYPE PIC X VALUE '0'.
10 ENQ-RT-R-HHHH PIC 9999 COMP VALUE ZERO.
10 ENQ-RT-R-ITEM PIC 9999 COMP VALUE ZERO.

---DELETE QUEUE TS QUEUE ID
05 ENQ-DQ-QUEUE-ID.
10 ENQ-DQ-CONSTANT PIC X(3) VALUE 'MQT'.
10 ENQ-DQ-TYPE PIC X VALUE 'D'.
10 ENQ-DQ-HHHH PIC 9999 VALUE ZERO.
10 ENQ-DQ-ITEM PIC 9999 COMP VALUE ZERO.

---ENQ FOR COMMUNICATION HANDLERS - SENDERS
05 ENQ-COMH-ID.
10 ENQ-COMH-CONSTANT PIC X(3) VALUE 'MQT'.
10 ENQ-COMH-ENTRY PIC 9(5) VALUE ZERO.

-----*
-----*
---OPEN PARM
01 WS-Q-NAME-AREA.
* COPY CMQODV.
*****
** FILE NAME: CMQODV **
** DESCRIPTIVE NAME: COBOL copy file for MQOD structure **
** VERSION 1.3.0 **
** FUNCTION: This file declares the MQOD structure, **
** which forms part of the IBM Message **
** Queue Interface (MQI). **
*****
** MQOD structure
10 MQOD.
** Structure identifier
15 MQOD-STRUCID PIC X(4) VALUE '0D '.
** Structure version number
15 MQOD-VERSION PIC S9(9) BINARY VALUE 1.
** Object type
15 MQOD-OBJECTTYPE PIC S9(9) BINARY VALUE 1.
** Object name
15 MQOD-OBJECTNAME PIC X(48) VALUE SPACES.
** Object queue manager name
15 MQOD-OBJECTQMGRNAME PIC X(48) VALUE SPACES.
** Dynamic queue name
15 MQOD-DYNAMICQNAME PIC X(48) VALUE '*'.
** Alternate user identifier
15 MQOD-ALTERNATEUSERID PIC X(12) VALUE SPACES.

---INQ
01 MQI-SELECTOR-COUNT.
05 WS-SELECTOR-COUNT PIC S9(8) COMP.
01 MQI-SELECTOR.
05 WS-SELECTOR PIC XXXX.

01 MQI-IN-ATTR-COUNT.
05 WS-IN-ATTR-COUNT PIC S9(8) COMP.
01 MQI-IN-ATTR.
05 WS-IN-ATTR PIC XXXX.

01 MQI-CHAR-ATTR-LENGTH.
05 WS-CHAR-ATTR-LENGTH PIC S9(8) COMP.
01 MQI-CHAR-ATTR.
05 WS-CHAR-ATTR PIC XXXX.

---PUT/GET PARM
01 WS-MSG-DESCRIPTOR.
* COPY CMQMDV.
*****
** FILE NAME: CMQMDV **

```

MQPECHO.Z

```

**                               **
** DESCRIPTIVE NAME: COBOL copy file for MQMD structure **
**                               **
** VERSION 1.3.0                 **
**                               **
** FUNCTION:                     ** This file declares the MQMD structure, **
**                               ** which forms part of the IBM Message **
**                               ** Queue Interface (MQI). **
**                               **
*****
** MQMD structure                **
** 10 MQMD.                      **
** Structure identifier          **
** 15 MQMD-STRUCID              ** PIC X(4) VALUE 'MD '. **
** Structure version number     **
** 15 MQMD-VERSION              ** PIC S9(9) BINARY VALUE 1. **
** Reserved                     **
** 15 MQMD-REPORT               ** PIC S9(9) BINARY VALUE 0. **
** Message type                 **
** 15 MQMD-MSGTYPE              ** PIC S9(9) BINARY VALUE 8. **
** Reserved                     **
** 15 MQMD-EXPIRY               ** PIC S9(9) BINARY VALUE -1. **
** Feedback code                **
** 15 MQMD-FEEDBACK             ** PIC S9(9) BINARY VALUE 0. **
** Data encoding                **
** 15 MQMD-ENCODING             ** PIC S9(9) BINARY VALUE 785. **
** Coded character set identifier **
** 15 MQMD-CODEDCHARSETID       ** PIC S9(9) BINARY VALUE 0. **
** Format name                   **
** 15 MQMD-FORMAT               ** PIC X(8) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-PRIORITY             ** PIC S9(9) BINARY VALUE 0. **
** Message persistence          **
** 15 MQMD-PERSISTENCE          ** PIC S9(9) BINARY VALUE 2. **
** Message identifier           **
** 15 MQMD-MSGID                ** PIC X(24) VALUE LOW-VALUES. **
** Correlation identifier       **
** 15 MQMD-CORRELID             ** PIC X(24) VALUE LOW-VALUES. **
** Reserved                     **
** 15 MQMD-BACKOUTCOUNT        ** PIC S9(9) BINARY VALUE 0. **
** Name of reply queue          **
** 15 MQMD-REPLYTOQ             ** PIC X(48) VALUE SPACES. **
** Name of reply queue manager **
** 15 MQMD-REPLYTOQMGR          ** PIC X(48) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-USERIDENTIFIER       ** PIC X(12) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-ACCOUNTINGTOKEN      ** PIC X(32) VALUE LOW-VALUES. **
** Reserved                     **
** 15 MQMD-APPLIDENTITYDATA     ** PIC X(32) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-PUTAPPLTYPE          ** PIC S9(9) BINARY VALUE 0. **
** Reserved                     **
** 15 MQMD-PUTAPPLNAME          ** PIC X(28) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-PUTDATE              ** PIC X(8) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-PUTTIME              ** PIC X(8) VALUE SPACES. **
** Reserved                     **
** 15 MQMD-APPLORIGINDATA       ** PIC X(4) VALUE SPACES. **
**                               **
01 WS-PUT-OPTIONS.
* COPY CMQPMOV.
*****
**                               **
** FILE NAME:                    ** CMQPMOV **
**                               **
** DESCRIPTIVE NAME: COBOL copy file for MQPMO structure **
**                               **
** VERSION 1.3.0                 **
**                               **
** FUNCTION:                     ** This file declares the MQPMO structure, **
**                               ** which forms part of the IBM Message **
**                               ** Queue Interface (MQI). **
**                               **
*****
** MQPMO structure                **
** 10 MQPMO.                      **
** Structure identifier          **
** 15 MQPMO-STRUCID              ** PIC X(4) VALUE 'PMO '. **
** Structure version number     **
** 15 MQPMO-VERSION              ** PIC S9(9) BINARY VALUE 1. **
** Options                      **
** 15 MQPMO-OPTIONS              ** PIC S9(9) BINARY VALUE 0. **
** Wait interval                 **
** 15 MQPMO-WAITINTERVAL         ** PIC S9(9) BINARY VALUE 0. **
** Signal                        **
** 15 MQPMO-SIGNAL1              ** PIC S9(9) BINARY VALUE 0. **
** Reserved                     **
** 15 MQPMO-SIGNAL2              ** PIC S9(9) BINARY VALUE 0. **
** Resolved name of destination queue **
** 15 MQPMO-RESOLVEDQNAME        ** PIC X(48) VALUE SPACES. **
**                               **
*****
* COMMON PARMS
01 WS-PARMS.
05 WS-HCONN-VALUE                USAGE POINTER.
05 WS-HOBJ-VALUE                 USAGE POINTER.
05 WS-PUT-HOBJ-VALUE             USAGE POINTER.
05 WS-CCODE-VALUE                PIC S9(8) COMP.
05 WS-RCODE-VALUE                PIC S9(8) COMP.
05 WS-QM-NAME-CONNECT            PIC X(48).
05 WS-Q-OPEN-OPTIONS-VALUE       PIC S9(8) COMP.
05 WS-DATA-LENGTH-USER           PIC S9(8) COMP.
05 WS-BUFFER-LENGTH              PIC S9(8) COMP.
**                               **
05 WS-BUFFER-AREA.
10 FILLER                        PIC X(8000).
**                               **
* COPY MQWEOWS.
*****
*Debugging eyecatcher information for end of WORKING-STORAGE.
*****
01 FILLER                        PIC X(16) VALUE
'====='.
01 WS-RWS-PROGRAM-NAME2          PIC X(8).
01 FILLER                        PIC X(16) VALUE
' Version V2.1.0'.
*****
*****
LINKAGE SECTION.
*****

```



```

01 DFHCOMMAREA.
05 FILLER PIC X(1000).

-----*
-----*
PROCEDURE DIVISION.
-----*

0000-MAIN-LINE.
MOVE 'MQPECHO' TO WS-RWS-PROGRAM-NAME1
                WS-RWS-PROGRAM-NAME2.
MOVE WHEN-COMPILED TO WS-RWS-WHEN-COMPILED.

*--INITIALIZE
PERFORM 1000-INITIALIZE
THRU 1000-EXIT.

*--CONNECT AND OPEN GET QUEUE
PERFORM 2000-CONNECT.
PERFORM 3100-GET-OPEN.

-----*
SET WS-MORE-DATA TO TRUE.
PERFORM
UNTIL (WS-NOMORE-DATA)

*--GET MESSAGE
PERFORM 3500-GET-MESSAGES

END-PERFORM.

*--CLOSE AND DISC
PERFORM 3900-GET-CLOSE.
PERFORM 5000-DISCONNECT.

-----*
0000-RETURN.
EXEC CICS RETURN
END-EXEC.

GOBACK.

-----*
1000-INITIALIZE.
-----*
* PURPOSE: SETUP DATA AREAS
-----*

*--GET ENVIRONMENT INFO
PERFORM 1015-GET-ENVIR-RECORD
THRU 1015-GET-ENVIR-EXIT.

*--SET UP ERROR AREA
PERFORM 1050-SET-ERROR-INFO.

*
*--CHECK IF QUEUE PRESENT
IF EIBCALEN < LENGTH OF MQTM
THEN
GO TO 0000-RETURN.

*--MOVE QUEUE NAME
MOVE DFHCOMMAREA TO MQTM.

-----*
1000-EXIT.
EXIT.

-----*
1015-GET-ENVIR-RECORD.
-----*
* PURPOSE: READ ENVIRONMENT RECORD
-----*

*--SET HANDLE
EXEC CICS HANDLE CONDITION
                QIDERR (9900-NO-ENVIR-SETUP)
                ITEMERR (9900-NO-ENVIR-SETUP)

END-EXEC.

*--READ ANCHOR FOR QM
MOVE LENGTH OF ENV-DEFINITION TO ENQ-ENVIR-TS-SIZE.

EXEC CICS READQ TS
                QUEUE (ENQ-ENVIR-TS-QID)
                INTO (ENV-DEFINITION)
                LENGTH (ENQ-ENVIR-TS-SIZE)
                ITEM (ENQ-ENVIR-TS-ITEM)
END-EXEC.

*--CHECK IF GOOD SIZE
IF LENGTH OF ENV-DEFINITION
NOT EQUAL ENQ-ENVIR-TS-SIZE
GO TO 9900-NO-ENVIR-SETUP
END-IF.

-----*
1015-GET-ENVIR-EXIT.
EXIT.

-----*
1050-SET-ERROR-INFO.
-----*
* PURPOSE: SET DEFAULT ERROR INFO
-----*

*--SET CSMT DATE AND TIME
EXEC CICS ASKTIME
                ABSTIME(WS-ABSTIME)
END-EXEC.

MOVE EIBTIME TO WS-UNPACK-TIME-9.
MOVE WS-TIME-HH TO WS-FORMAT-TIME-HH
MOVE WS-TIME-MM TO WS-FORMAT-TIME-MM.
MOVE WS-TIME-SS TO WS-FORMAT-TIME-SS.

EXEC CICS FORMATTIME
                ABSTIME (WS-ABSTIME)
                MMDDYY (WS-FORMATTED-DATE)
                DATESEP ('/')
END-EXEC.

*
EXEC CICS FORMATTIME
                ABSTIME(WS-ABSTIME)
                YYMMDD (WS-DATE-YYMMDD)
END-EXEC.

*-- --SET CENTURY
IF WS-DATE-YY > 50
MOVE 19 TO WS-DATE-CC
ELSE
MOVE 20 TO WS-DATE-CC
END-IF.

*--SET COMMON ERROR INFO
MOVE ZERO TO ERR-CODE.
MOVE ENV-II-LINK-ECHO TO ERR-PROGRAM.

-----*
-----*
2000-CONNECT.
-----*
* PURPOSE: CONNECT
-----*

*--MQCONNECT TO QM
MOVE 'CONNECT' TO WS-FUNCTION.
MOVE SPACES TO WS-QM-NAME-CONNECT.
MOVE MQCC-OK TO WS-CCODE-VALUE.
MOVE MQRC-NONE TO WS-RCODE-VALUE.
SET WS-HCONN-VALUE TO NULL.
CALL 'MQCONN' USING WS-QM-NAME
                WS-HCONN-VALUE
                WS-CCODE-VALUE
                WS-RCODE-VALUE.

*
IF WS-CCODE-VALUE NOT EQUAL ZERO
GO TO 9900-ERR-DISPLAY
END-IF.

-----*
-----*
3100-GET-OPEN.
-----*
* PURPOSE: OPEN

```

MQPECHO.Z

```

-----*
*--MQOPEN  QUEUE TO QM
  MOVE 'OPEN'  TO WS-FUNCTION.
  MOVE MQ00-INPUT-SHARED TO WS-Q-OPEN-OPTIONS-VALUE.
  MOVE SPACES  TO MQ0D-OBJECTQMGRNAME.
  MOVE MQI-PROC-LOCAL-QUEUE-NAME
    TO MQ0D-OBJECTNAME.
  MOVE MQCC-OK  TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  SET WS-HOBJ-VALUE TO NULL.
  CALL 'MQOPEN' USING WS-HCONN-VALUE
    WS-Q-NAME-AREA
    WS-Q-OPEN-OPTIONS-VALUE
    WS-HOBJ-VALUE
    WS-CCODE-VALUE
    WS-RCODE-VALUE.

*
  IF WS-CCODE-VALUE NOT EQUAL ZERO
    GO TO 9900-ERR-DISPLAY
  END-IF.

*
-----*
3500-GET-MESSAGES.
-----*
*--MQGET TO QUEUE TO QM
  MOVE 'GET'  TO WS-FUNCTION.
  MOVE MQCC-OK  TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  MOVE LOW-VALUES TO MQMD-MSGID
    MQMD-CORRELID.
  MOVE LENGTH OF WS-BUFFER-AREA TO WS-BUFFER-LENGTH.
  MOVE MQGMO-ACCEPT-TRUNCATED-MSG
    TO MQGMO-OPTIONS.

  CALL 'MQGET' USING WS-HCONN-VALUE
    WS-HOBJ-VALUE
    WS-MSG-DESCRIPTOR
    WS-GET-OPTIONS
    WS-BUFFER-LENGTH
    WS-BUFFER-AREA
    WS-DATA-LENGTH
    WS-CCODE-VALUE
    WS-RCODE-VALUE.

*
*
  IF (WS-CCODE-VALUE NOT EQUAL ZERO)
  AND (WS-RCODE-VALUE NOT EQUAL MQRC-TRUNCATED-MSG-ACCEPTED)
  IF WS-RCODE-VALUE EQUAL MQRC-NO-MSG-AVAILABLE
    SET WS-NOMORE-DATA TO TRUE
  ELSE
    GO TO 9900-ERR-DISPLAY
  END-IF
  END-IF.

*--SEND QUEUE RECORDS
  IF WS-MORE-DATA
*-- --FIRST CHECK IF ANY REPLY
  PERFORM 4000-PUT1-MESSAGES
    THRU 4000-EXIT
  END-IF.

*--SYNCPPOINT
  EXEC CICS SYNCPPOINT
  END-EXEC.

*
*
-----*
3900-GET-CLOSE.
-----*
* PURPOSE: CLOSE
-----*
*--MQCLOSE QUEUE TO QM
  MOVE 'CLOSE' TO WS-FUNCTION.
  MOVE ZERO TO WS-Q-OPEN-OPTIONS-VALUE.
  MOVE MQCC-OK TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.

  CALL 'MQCLOSE' USING WS-HCONN-VALUE
    WS-HOBJ-VALUE
    WS-Q-OPEN-OPTIONS-VALUE
    WS-CCODE-VALUE
    WS-RCODE-VALUE.

  IF WS-CCODE-VALUE NOT EQUAL ZERO
    GO TO 9900-ERR-DISPLAY
  END-IF.

-----*
4000-PUT1-MESSAGES.
-----*
* PURPOSE: PUT1
-----*
*--MQPUT1  QUEUE TO QM
  MOVE 'PUT1' TO WS-FUNCTION.
  MOVE MQ00-OUTPUT TO WS-Q-OPEN-OPTIONS-VALUE.
  IF MQMD-REPLYTOQMGR EQUAL SPACES OR LOW-VALUES
    MOVE SPACES TO MQ0D-OBJECTQMGRNAME
  ELSE
    MOVE MQMD-REPLYTOQMGR
      TO MQ0D-OBJECTQMGRNAME
  END-IF.

*
*--IF NOT REPLY QUEUE - SET DEFAULT
  IF MQMD-REPLYTOQ EQUAL SPACES OR LOW-VALUES
    MOVE WS-R-Q-NAME
      TO MQ0D-OBJECTNAME
  ELSE
    MOVE MQMD-REPLYTOQ
      TO MQ0D-OBJECTNAME
  END-IF.

*
  MOVE MQCC-OK  TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  MOVE WS-DATA-LENGTH-USER TO WS-BUFFER-LENGTH.
  CALL 'MQPUT1' USING WS-HCONN-VALUE
    WS-Q-NAME-AREA
    WS-MSG-DESCRIPTOR
    WS-PUT-OPTIONS
    WS-BUFFER-LENGTH
    WS-BUFFER-AREA
    WS-CCODE-VALUE
    WS-RCODE-VALUE.

*
  IF WS-CCODE-VALUE NOT EQUAL ZERO
    GO TO 9900-ERR-DISPLAY
  END-IF.

-----*
4000-EXIT.
  EXIT.

-----*
5000-DISCONNECT.
-----*
* PURPOSE: DISCON
-----*
*--MQDISC FROM QM
  MOVE 'DISCONN' TO WS-FUNCTION.
  MOVE MQCC-OK TO WS-CCODE-VALUE.
  MOVE MQRC-NONE TO WS-RCODE-VALUE.
  CALL 'MQDISC' USING WS-HCONN-VALUE
    WS-CCODE-VALUE
    WS-RCODE-VALUE.

*
-----*
9900-ERR-DISPLAY.
-----*
*--ERROR IN "MQ" VERB
*
  MOVE ERR-INT-RETURN-ERROR TO ERR-CODE.

```

```

MOVE MQI-PROC-LOCAL-QUEUE-NAME TO ERR-QUEUE.
PERFORM 9999-CONVERT-ERROR-INFO.

*--WRITE ERROR
PERFORM 9999-ERROR-WRITE.
*
*--ALWAYS DISCONNECT (NOTE NO ERROR CHECKING IN DISCONNECT)

*--SYNCPOINT - ROLLBACK
EXEC CICS SYNCPOINT
      ROLLBACK
END-EXEC.
*
PERFORM 5000-DISCONNECT.
GO TO 0000-RETURN.
*

-----*
9900-CICS-PGMIDERR.
-----*

*--SET MESSAGE AND CODE
MOVE ERR-CICS-PGMIDERR TO ERR-CODE.

```

```

*--CONVERT ERROR CODE
PERFORM 9999-CONVERT-ERROR-INFO.

*--WRITE ERROR
PERFORM 9999-ERROR-WRITE.

*--RETURN
GO TO 0000-RETURN.

-----*
* ERROR PROCESSING
-----*
* COPY MQIERRCD.
/INCLUDE MQIERRCD

*--ADDED CODE FOR ABEND CONDITION

*--RETURN
GO TO 0000-RETURN.

9900-NO-ENVIR-SETUP.
GO TO 0000-RETURN.

```

Appendix F. Example configuration - MQSeries for VSE/ESA Version 2.1.0

This appendix gives an example of how to set up communication links from MQSeries for VSE/ESA to MQSeries products on the following platforms:

- OS/2
- Windows NT
- AIX
- HP-UX
- AT&T GIS UNIX²
- Sun Solaris
- OS/400®
- MVS/ESA without CICS

It describes the parameters needed for an LU 6.2 and TCP/IP connection. Once the connection is established, you need to define some channels to complete the configuration. This is described in “MQSeries for VSE/ESA configuration” on page 190.

Configuration parameters for an LU 6.2 connection

Table 8 on page 186 presents a worksheet listing all the parameters needed to set up communication from VSE/ESA to one of the other MQSeries platforms. The worksheet shows examples of the parameters, which have been tested in a working environment, and leaves space for you to fill in your own values. An explanation of the parameter names follows the worksheet. Use the worksheet in this chapter in conjunction with the worksheet in the *MQSeries Intercommunication* book for the platform to which you are connecting.

Configuration worksheet

Use the following worksheet to record the values you will use for this configuration. Where numbers appear in the Reference column they indicate that the value must match that in the appropriate worksheet in the *MQSeries Intercommunication* book. The examples that follow in this chapter refer back to the values in the ID column of this table. The entries in the Parameter Name column are explained in “Explanation of terms” on page 187.

² This platform has become NCR UNIX SVR4 MP-RAS, R3.0

Table 8 (Page 1 of 2). Configuration worksheet for VSE/ESA using APPC

ID	Parameter Name	Reference	Example Used	User Value
Definition for local node				
1	Network ID		NETID	
2	Node name		VSEPU	
3	Local LU name		VSELU	
4	Local Transaction Program name		MQ01	MQ01
5	LAN destination address		400074511092	
Connection to an OS/2 system				
The values in this section of the table must match those used in the table for OS/2 in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		OS2	
7	Group name		EXAMPLE	
8	Session name		OS2SESS	
9	Netname	6	OS2LU	
Connection to a Windows NT system				
The values in this section of the table must match those used in the table for Windows NT in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		WNT	
7	Group name		EXAMPLE	
8	Session name		WNTSESS	
9	Netname	5	WINNTLU	
Connection to an AIX system				
The values in this section of the table must match those used in the table for AIX in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		AIX	
7	Group name		EXAMPLE	
8	Session name		AIXSESS	
9	Netname	4	AIXLU	
Connection to an HP-UX system				
The values in this section of the table must match those used in the table for HP-UX in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		HPUX	
7	Group name		EXAMPLE	
8	Session name		HPUXSESS	
9	Netname	5	HPUXLU	
Connection to an AT&T GIS UNIX system				
The values in this section of the table must match those used in the table for GIS UNIX in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		GIS	
7	Group name		EXAMPLE	
8	Session name		GISSESS	
9	Netname	4	GISLU	

ID	Parameter Name	Reference	Example Used	User Value
Table 8 (Page 2 of 2). Configuration worksheet for VSE/ESA using APPC				
Connection to a Sun Solaris system				
The values in this section of the table must match those used in the table for Sun Solaris in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		SOL	
7	Group name		EXAMPLE	
8	Session name		SOLSESS	
9	Netname	5	SOLARLU	
Connection to an AS/400 system				
The values in this section of the table must match those used in the table for AS/400 in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		AS4	
7	Group name		EXAMPLE	
8	Session name		AS4SESS	
9	Netname	3	AS400LU	
Connection to an MVS/ESA system without CICS				
The values in this section of the table must match those used in the table for MVS/ESA in the <i>MQSeries Intercommunication</i> book, as indicated.				
6	Connection name		MVS	
7	Group name		EXAMPLE	
8	Session name		MVSSESS	
9	Netname	4	MVSLU	

Explanation of terms

1 Network ID

This is the unique ID of the network to which you are connected. Your system administrator will tell you this value.

2 Node name

This is the name of the SSCP which owns the CICS/VSE region.

3 Local LU name

This is the unique VTAM APPLID of this CICS/VSE region.

4 Transaction Program name

MQSeries applications trying to converse with this queue manager will specify a transaction name for the program to be run at the receiving end. This will have been defined on the channel definition at the sender. MQSeries for VSE/ESA uses a name of MQ01.

5 LAN destination address

This is the LAN destination address that your partner nodes will use to communicate with this host. It is usually the address of the 3745 on the same LAN as the partner node.

6 Connection name

This is a 4-character name by which each connection will be individually known in CICS RDO.

7 Group name

You choose your own 8-character name for this value. Your system may already have a group defined for connections to partner nodes. Your system administrator will give you a value to use.

8 Session name

This is an 8-character name by which each session will be individually known. For clarity we use the connection name, concatenated with 'SESS'.

9 Netname

This is the LU name of the MQSeries queue manager on the system with which you are setting up communication.

Establishing an LU 6.2 connection

This example is for a connection to an OS/2 system. The steps are the same whatever platform you are using; change the values as appropriate.

Defining a connection

1. At a CICS command line type `CEDA DEF CONN(connection name) 6 GROUP(group name) 7`. For example:
`CEDA DEF CONN(OS2) GROUP(EXAMPLE)`
2. Press Enter to define a connection to CICS.

```
DEF CONN(OS2) GROUP(EXAMPLE)
OVERTYPE TO MODIFY
CEDA DEFINE
Connection : OS2
Group      : EXAMPLE
Description ==>
CONNECTION IDENTIFIERS
Netname    ==> OS2LU
INDsys     ==>
REMOTE ATTRIBUTES
REMOTESystem ==>
REMOTENAME ==>
CONNECTION PROPERTIES
ACccessmethod ==> Vtam      Vtam | IRc | INdirect | Xm
Protocol      ==> Appc      Appc | Lu61
SIngleless   ==> No        No | Yes
DATAstream    ==> User      User | 3270 | SCs | STRfield | Lms
RECORDformat ==> U          U | Vb
OPERATIONAL PROPERTIES
+ AUTOconnect ==> Yes      No | Yes | All
I New group EXAMPLE created.

DEFINE SUCCESSFUL          TIME: 16.49.30 DATE: 96.054
PF 1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

3. On the panel change the **Netname** field in the CONNECTION IDENTIFIERS section to be the LU name (**9**) of the target system.
4. In the CONNECTION PROPERTIES section set the **ACccessmethod** field to Vtam and the **Protocol** to Appc.
5. Press Enter to make the change.

Defining a session

1. At a CICS command line type `CEDA DEF SESS(session name) 8 GROUP(group name) 7`. For example:
`CEDA DEF SESS(OS2SESS) GROUP(EXAMPLE)`
2. Press Enter to define a session for the connection.

```
DEF SESS(OS2SESS) GROUP(EXAMPLE)
OVERTYPE TO MODIFY
CEDA DEFINE
Sessions ==> OS2SESS
Group    ==> EXAMPLE
Description ==>
SESSION IDENTIFIERS
Connection ==> OS2
SESSName  ==>
NETNameq  ==>
MOnename  ==> #INTER
SESSION PROPERTIES
Protocol  ==> Appc      Appc | Lu61
Maximum  ==> 008 , 004  0-999
RECEIVEPfx ==>
RECEIVECount ==> 1-999
SENDPfx   ==>
SENDCount  ==> 1-999
SENDSize  ==> 04096    1-30720
+ RECEIVESize ==> 04096  1-30720
S CONNECTION MUST BE SPECIFIED.

TIME: 14.23.19 DATE: 96.054
PF 1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

3. In the SESSION IDENTIFIERS section of the panel specify the Connection name (**6**) in the **Connection** field and set the **MOnename** to #INTER.
4. In the SESSION PROPERTIES section set the **Protocol** to Appc and the **MAXimum** field to 008 , 004.

Installing the new group definition

1. At a CICS command line type `CEDA INS GROUP(group name) 7`.
2. Press Enter to install the new group definition.

Note: If this connection group is already in use you may get severe errors reported. If this happens, take the existing connections out of service, retry the above group installation, and then set the connections in service using the following commands:

- a. `CEMT I CONN`
- b. `CEMT S CONN(*) OUTS`
- c. `CEDA INS GROUP(group name)`
- d. `CEMT S CONN(*) INS`

What next?

The LU 6.2 connection is now established. You are ready to complete the configuration. Go to "MQSeries for VSE/ESA configuration" on page 190.

Establishing a TCP/IP connection

TCP/IP connections do not require the configuration of additional profiles as does the LU 6.2 protocol. Instead, MQSeries for VSE/ESA processes the MQSeries listener program during MQSeries startup.

The MQSeries listener program waits for remote TCP/IP connection requests. As these are received, the listener starts the receiver MCA to process the remote connection. When the remote connection is received from a client program, the receiver MCA starts the MQSeries server program.

Note: There is one MQSeries server process for each client connection.

Provided that the MQ Listener is active and TCP/IP is active in a VSE partition, TCP/IP connections can be established.

MQSeries for VSE/ESA configuration

Configuring MQSeries for VSE/ESA involves the following tasks:

- Configuring channels
- Defining a local queue
- Defining a remote queue
- Defining a sender channel
- Defining a receiver channel

Note: MQSeries for VSE/ESA does not understand the format of the MQSeries channel ping command. The only way to verify your MQSeries definitions is to start the channels and put messages onto remote queues.

Configuring channels

Examples are given for connecting MQSeries for VSE/ESA and MQSeries for OS/2 Warp. If you wish connect to another MQSeries platform use the appropriate set of values from the table in place of those for OS/2.

Note: The words in **bold** are user-specified and reflect the names of MQSeries objects used throughout these examples. If you change the names used here, ensure that you also change the other references made to these objects throughout this book. All others are keywords and should be entered as shown.

Refer to the sections “Defining a local queue” on page 194 and “Defining a remote queue” on page 195 for details of how to create queue definitions, and “Defining a SNA LU6.2 sender channel” on page 195 and “Defining a SNA LU6.2 receiver channel” on page 196 for details of how to create channels.

Table 9 (Page 1 of 3). Configuration worksheet for MQSeries for VSE/ESA

ID	Parameter Name	Reference	Example Used	User Value
<i>Definition for local node</i>				
A	Queue Manager Name		VSEP	
B	Local queue name		VSE.LOCALQ	

ID	Parameter Name	Reference	Example Used	User Value
Table 9 (Page 2 of 3). Configuration worksheet for MQSeries for VSE/ESA				
Connection to MQSeries for OS/2 Warp				
The values in this section of the table must match those used in the worksheet table for OS/2 in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name	A	OS2	
D	Remote queue name		OS2.REMOTEQ	
E	Queue name at remote system	B	OS2.LOCALQ	
F	Transmission queue name		OS2	
G	Sender channel name		VSE.OS2.SNA	
I	Receiver channel name	G	OS2.VSE.SNA	
Connection to MQSeries for Windows NT				
The values in this section of the table must match those used in the worksheet table for Windows NT in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name	A	WINNT	
D	Remote queue name		WINNT.REMOTEQ	
E	Queue name at remote system	B	WINNT.LOCALQ	
F	Transmission queue name		WINNT	
G	Sender channel name		VSE.WINNT.SNA	
I	Receiver channel name	G	WINNT.VSE.SNA	
Connection to MQSeries for AIX				
The values in this section of the table must match those used in the worksheet table for AIX in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name		AIX	
D	Remote queue name		AIX.REMOTEQ	
E	Queue name at remote system	B	AIX.LOCALQ	
F	Transmission queue name		AIX	
G	Sender channel name		VSE.AIX.SNA	
I	Receiver channel name	G	AIX.VSE.SNA	
Connection to MQSeries for HP-UX				
The values in this section of the table must match those used in the worksheet table for HP-UX in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name		HPUX	
D	Remote queue name		HPUX.REMOTEQ	
E	Queue name at remote system	B	HPUX.LOCALQ	
F	Transmission queue name		HPUX	
G	Sender channel name		VSE.HPUX.SNA	
I	Receiver channel name	G	HPUX.VSE.SNA	
Connection to MQSeries for AT&T GIS UNIX				
The values in this section of the table must match those used in the worksheet table for GIS UNIX in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name		GIS	
D	Remote queue name		GIS.REMOTEQ	
E	Queue name at remote system	B	GIS.LOCALQ	
F	Transmission queue name		GIS	
G	Sender channel name		VSE.GIS.SNA	
I	Receiver channel name	G	GIS.VSE.SNA	

VSE/ESA configuration

ID	Parameter Name	Reference	Example Used	User Value
Connection to MQSeries for Sun Solaris				
The values in this section of the table must match those used in the worksheet table for Sun Solaris in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name		SOLARIS	
D	Remote queue name		SOLARIS.REMOTEQ	
E	Queue name at remote system	B	SOLARIS.LOCALQ	
F	Transmission queue name		SOLARIS	
G	Sender channel name		VSE.SOLARIS.SNA	
I	Receiver channel name	G	SOLARIS.VSE.SNA	
Connection to MQSeries for AS/400				
The values in this section of the table must match those used in the worksheet table for AS/400 in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name		AS400	
D	Remote queue name		AS400.REMOTEQ	
E	Queue name at remote system	B	AS400.LOCALQ	
F	Transmission queue name		AS400	
G	Sender channel name		VSE.AS400.SNA	
I	Receiver channel name	G	AS400.VSE.SNA	
Connection to MQSeries for MVS/ESA without CICS				
The values in this section of the table must match those used in the worksheet table for MVS/ESA in the <i>MQSeries Intercommunication</i> book, as indicated.				
C	Remote queue manager name		MVS	
D	Remote queue name		MVS.REMOTEQ	
E	Queue name at remote system	B	MVS.LOCALQ	
F	Transmission queue name		MVS	
G	Sender channel name		VSE.MVS.SNA	
I	Receiver channel name	G	MVS.VSE.SNA	

For TCP/IP, the sender channel name **G** and the receiver channel name **I**, in the preceding table, can be VSE.sys.tcp and sys.VSE.TCP respectively.

In both cases sys represents the remote system name, for example, OS2. Therefore, in this case, **G** becomes VSE.OS2.TCP and **I** becomes OS2.VSE.TCP.

MQSeries for VSE/ESA sender-channel definitions

```
Local Queue
  Object Type : L
  Object Name : OS2 F
  Usage Mode: T (Transmission)
```

```
Remote Queue
  Object Type : R
  Object Name : OS2.REMOTEQ D
  Remote QUEUE Name : OS2.LOCALQ E
  Remote QM Name : OS2 C
  Transmission Name : OS2 F
```

```
Sender Channel
```

Channel name : **VSE.OS2.SNA** **G**
Channel type : S (Sender)
Transmission queue name : **OS2** **F**
Remote Task ID : MQTP
Connection name : **OS2** **G**

MQSeries for VSE/ESA receiver-channel definitions

Local Queue

Object type : QLOCAL
Object Name : **VSE.LOCALQ** **B**
Usage Mode : N (Normal)

Receiver Channel

Channel name : **OS2.VSE.SNA** **I**
Channel type : R (Receiver)

Defining a local queue

1. Run the MQSeries master terminal transaction MQMT.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:50:25      *** Master Terminal Main Menu ***          VSE1
MQMMTP                                               A004

                SYSTEM IS ACTIVE

                1. Configuration
                2. Operations
                3. Monitoring
                4. Browse Queue Records

                Option:

Function completed - please enter a new request.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
CLEAR/PF3 = Exit                          ENTER=Select
    
```

2. Select option 1 to configure.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:52:21      *** Configuration Main Menu ***          VSE1
MQMCFG                                               A004

                SYSTEM IS ACTIVE

                Maintenance Options :
                1. Global System Definition
                2. Queue Definitions
                3. Channel Definitions

                Display Options :
                4. Global System Definition
                5. Queue Definitions
                6. Channel Definitions

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process      PF2 = Main Menu      PF3 = Quit
    
```

3. Select option 2 to work with queue definitions.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:55:12      Queue Main Options                      VSE1
MQMMQUE                                               A004

                SYSTEM IS ACTIVE

                Default Q Manager : VSEP

                Object Type: L      L=Local Q, R=Remote Q, AQ=Alias Queue,
                                AM=Alias Manager,
                                AR=Alias Reply Q

                Object Name: VSE.LOCALQ

                ENTER NEEDED INFORMATION.

                PF2=Main Config PF3 = Quit PF4/ENTER = Read PF5 = Add PF6 = Update
                                PF9 = List PF11= Reorg. PF12= Delete
    
```

4. Select an Object type of L and specify the name of the queue.

5. Press PF5.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:56:10      Queue Definition Record                VSE1
MQMMQUE      QM - VSEP                                A004

                LOCAL QUEUE DEFINITION

                Object Name . . . . . : VSE.LOCALQ
                Description line 1 . . . . . :
                Description line 2 . . . . . :

                Put Enabled . . . . . : Y      Y=Yes, N=No
                Get Enabled . . . . . : Y      Y=Yes, N=No

                Default Inbound status . . : A      Outbound . . : A      A=Active,I=Inactive

                Dual Update Queue . . . . . :

                Automatic Reorganize (Y/N) : N

                Record being added - Press ADD key again.

                PF2=Main Config PF3 = Quit PF4/ENTER = Read PF5 = Add PF6 = Update
                                PF9 = List PF10= Queue PF11= Reorg. PF12= Delete
    
```

6. Press PF5 again.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:57:26      Queue Extended Definition              VSE1
MQMMQUE      QM - VSEP                                A004

                Object Name . . . . . : VSE.LOCALQ
                Physical Queue Information

                Usage Mode . . . . . : N      N=Normal, T=Transmission
                Share Mode . . . . . : Y      Y=Yes, N=No
                Physical File Name . . . . . : ** FILE NOT DEFINED

                Maximum Values
                Maximum Q Depth . . . . . : 01000000      Global Lock Entries . . : 00001000
                Maximum Message Length . . : 01000000      Local Lock Entries . . : 00001000
                Maximum Concurrent Accesses: 00000100      Checkpoint Threshold : 1000

                Trigger Information
                Trigger Enable . . . . . : N      Y=yes, N=No
                Trigger Type . . . . . : F=First, E=Every
                Maximum Trigger Starts . . : 0001
                Allow Restart of Trigger . . : N      Y=Yes, N=No
                Trans ID :                               Term ID:
                Program ID :                             Channel Name:

                **** File not found ****
                PF2=Main Config PF3 = Quit PF4/ENTER = Read PF5 = Add PF6 = Update
                                PF9 = List PF10= Queue PF11= Reorg. PF12= Delete
    
```

7. Specify the name of a CICS file to store messages for this queue.

8. If you are creating a transmission queue, specify a **Usage Mode** of T, a **Program ID** of MQPSEND, and a **Channel Name** < **G** >.

For a normal queue specify a **Usage Mode** of N.

9. Press PF5 again.

Defining a remote queue

1. Run the MQSeries master terminal transaction MQMT.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:50:25      *** Master Terminal Main Menu ***          VSE1
MQMMTP                                               A004

                SYSTEM IS ACTIVE

                1. Configuration
                2. Operations
                3. Monitoring
                4. Browse Queue Records

                Option:

Function completed - please enter a new request.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
CLEAR/PF3 = Exit                          ENTER=Select
    
```

2. Select option 1 to configure.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:52:21      *** Configuration Main Menu ***          VSE1
MQMMCFG                                               A004

                SYSTEM IS ACTIVE

                Maintenance Options :
                1. Global System Definition
                2. Queue Definitions
                3. Channel Definitions

                Display Options :
                4. Global System Definition
                5. Queue Definitions
                6. Channel Definitions

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process                          PF2 = Main Menu      PF3 = Quit
    
```

3. Select option 2 to work with queue definitions.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:59:30      Queue Main Options                          VSE1
MQMMQUE                                               A004

                SYSTEM IS ACTIVE

                Default Q Manager : VSEP

                Object Type: R      L=Local Q, R=Remote Q, AQ=Alias Queue,
                                AM=Alias Manager,
                                AR=Alias Reply Q

                Object Name: OS2.REMOTEQ

                ENTER NEEDED INFORMATION.

                PF2=Main Config PF3 = Quit PF4/ENTER = Read  PF5 = Add      PF6 = Update
                                PF9 = List  PF11= Reorg.    PF12= Delete
    
```

4. Select an **Object type** of **R** and specify the name of the queue.
5. Press PF5.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
20:00:25      Queue Definition Record                    VSE1
MQMMQUE       QM - VSEP                                    A004

                REMOTE QUEUE DEFINITION

                Object Name. . . . . : OS2.REMOTEQ
                Description line 1 . . . . . :
                Description line 2 . . . . . :

                Put Enabled . . . . . : Y      Y=Yes, N=No
                Get Enabled . . . . . : Y      Y=Yes, N=No

                Remote Queue Name . . . . . : OS2.LOCALQ
                Remote QM Name. . . . . : OS2
                Transmission Q Name . . . . . : OS2

                Record being added - Press ADD key again.

                PF2=Main Config PF3 = Quit  PF4/ENTER = Read  PF5 = Add      PF6 = Update
                                PF9 = List  PF10= Queue    PF11= Reorg.    PF12= Delete
    
```

6. Specify a remote queue name, remote queue manager name, and transmission queue name.
7. Press PF5.

Defining a SNA LU6.2 sender channel

1. Run the MQSeries master terminal transaction MQMT.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:50:25      *** Master Terminal Main Menu ***          VSE1
MQMMTP                                               A004

                SYSTEM IS ACTIVE

                1. Configuration
                2. Operations
                3. Monitoring
                4. Browse Queue Records

                Option:

Function completed - please enter a new request.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
CLEAR/PF3 = Exit                          ENTER=Select
    
```

2. Select option 1 to configure.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:52:21      *** Configuration Main Menu ***          VSE1
MQMMCFG                                               A004

                SYSTEM IS ACTIVE

                Maintenance Options :
                1. Global System Definition
                2. Queue Definitions
                3. Channel Definitions

                Display Options :
                4. Global System Definition
                5. Queue Definitions
                6. Channel Definitions

                Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process                          PF2 = Main Menu      PF3 = Quit
    
```

VSE/ESA configuration

- Select option 3 to work with channel definitions.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
20:01:18                Channel Record      DISPLAY      MCHN
MQMMCHN      Last Check Point      Last Update      A004
MSN          Time      Interv      Create Date
Name : VSEP.OSZ.SNA
Protocol : L (L/T) Port :          Type : S (S/R/C)
Partner : OSZ

Allocation Retries      Get Retries
Number of Retries: 00000000      Number of Retries : 00000000
Delay Time - fast: 00000000      Delay Time          : 00000010
Delay Time - slow: 00000000

Max Messages per Batch : 5          Max Transmission Size : 032000
Message Sequence Wrap : 999999      Max Message Size      : 008192

Mess Seq Req(Y/N): Y      Convers Cap (Y/N): Y      Split Mssg(Y/N): N

Transmission Queue Name : OSZ
TP Name: AMQRCS6A
Checkpoint Values:      Frequency: 0000      Time Span: 0000
Enable(Y/N) Y      Dead Letter Store(Y/N) Y
Enter Channel Name.
PF2 =Menu PF3 =Quit PF4 =Read PF5 =Add PF6=Update PF9 =List PF12 =Delete
    
```

- Complete the parameter fields as indicated, specifically the fields **Channel name**< **G** >, **Channel type**, **Connection ID**, **Remote task ID**, and **Transmit queue name**< **F** >.

All other parameters can be entered as shown.

Note that the default value for **sequence number wrap** is 999999, whereas for Version 2 MQSeries products, this value defaults to 999999999.

- Press PF5.

Defining a SNA LU6.2 receiver channel

- Run the MQSeries master terminal transaction MQMT.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:50:25                *** Master Terminal Main Menu ***      VSE1
MQMMTP                A004

SYSTEM IS ACTIVE

1. Configuration
2. Operations
3. Monitoring
4. Browse Queue Records

Option:

Function completed - please enter a new request.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
CLEAR/PF3 = Exit                          ENTER=Select
    
```

- Select option 1 to configure.

```

08/18/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
19:52:21                *** Configuration Main Menu ***      VSE1
MQMMCFG                A004

SYSTEM IS ACTIVE

Maintenance Options :
1. Global System Definition
2. Queue Definitions
3. Channel Definitions

Display Options :
4. Global System Definition
5. Queue Definitions
6. Channel Definitions

Option:

Please enter one of the options listed.
5686-A06 (C) Copyright IBM Corp. 1998      All Rights Reserved.
ENTER = Process      PF2 = Main Menu      PF3 = Quit
    
```

- Select option 3 to work with channel definitions.

```

08/19/1998      IBM MQSeries for VSE/ESA Version 2.1.0      IYBPZS01
07:29:03                Channel Record      DISPLAY      MCHN
MQMMCHN      Last Check Point      Last Update      19980805
MSN 00000149 Time 17:52:32 Interv 000000 Create Date 19980528
Name : OSZ.VSE.SNA
Protocol : L (L/T) Port : 0000 Type : R (S/R/C)
Partner :

Allocation Retries      Get Retries
Number of Retries: 00000000      Number of Retries : 00000000
Delay Time - fast: 00000000      Delay Time          : 00000000
Delay Time - slow: 00000000

Max Messages per Batch : 000001      Max Transmission Size : 032000
Message Sequence Wrap : 999999      Max Message Size      : 008192

Mess Seq Req(Y/N): Y      Convers Cap (Y/N): Y      Split Mssg(Y/N): N

Transmission Queue Name :
TP Name:
Checkpoint Values:      Frequency: 0000      Time Span: 0000
Enable(Y/N) Y      Dead Letter Store(Y/N) Y
Channel record displayed.
PF2 =Menu PF3 =Quit PF4 =Read PF5 =Add PF6=Update PF9 =List PF12 =Delete
    
```

- Complete the parameter fields as indicated, specifically the field **Channel name**< **L** >.

All other parameters can be entered as shown.

- Press PF5.

Defining a TCP/IP sender channel

To define a TCP/IP sender channel, carry out the following procedure:

1. Run the MQSeries master terminal transaction MQMT.
2. Select option 1 to configure.
3. Select option 3 to work with channel definitions. The screen shown in Figure 45 is displayed:

```

07/16/1998          IBM MQSeries for VSE/ESA Version 2.1.0          IYBPZS01
08:03:53           Channel Record          DISPLAY          MCHN
MQMMCHN           Last Check Point          Last Update 00000000    A005
MSN 00000002     Time 07:10:22   Interv 000000   Create Date 19980528
Name : SD01_TCP_VSEP
Protocol : T (L/T) Port : 1414   Type : S (S/R/C)
Partner :

      Allocation Retries          Get Retries
Number of Retries: 00000000      Number of Retries : 00000000
Delay Time - fast: 00000000      Delay Time         : 00000000
Delay Time - slow: 00000000

Max Messages per Batch : 000001      Max Transmission Size : 032000
Message Sequence Wrap : 999999      Max Message Size     : 008192

Mess Seq Req(Y/N): Y   Convers Cap (Y/N): Y   Split Mssg(Y/N): N

Transmission Queue Name :
TP Name:
Checkpoint Values:      Frequency: 0000   Time Span: 0000
Enable(Y/N) Y          Dead Letter Store(Y/N) N
Channel record displayed.
PF2 =Menu PF3 =Quit PF4 =Read PF5 =Add PF6=Update PF9 =List PF12 =Delete

```

Figure 45. Channel configuration panel

4. Complete the parameter fields as follows:
 - Channel name – **G** on the configuration worksheet.
 - Partner – should contain the IP address of the remote host, for example, 1.20.33.444.
 - Port – the port number must match the port number configured for the remote host. This is configured in the global system definition of the remote host. The default port number for MQSeries for VSE/ESA is 1414.
 - Transmission queue name – **F** on the configuration worksheet.
 - Protocol – enter T for TCP/IP
 - Channel type – enter S for sender

Notes:

- a. The TP Name is not used by TCP/IP channels.
 - b. Ensure that the parameter field values match the values of the receiver channel definition of the same name on the remote host.
5. Press PF5 (Add) to add the new channel definition.

Defining a TCP/IP receiver channel

To define a TCP/IP receiver channel, carry out the following procedure:

1. Run the MQSeries master terminal transaction MQMT.
2. Select option 1 to configure.
3. Select option 3 to work with channel definitions. The screen shown in Figure 45 on page 197 is displayed.
4. Complete the parameter fields as follows:
 - Channel name – **G** on the configuration worksheet.
 - Protocol – enter T for TCP/IP
 - Channel type – enter R for receiver.

Notes:

- a. The Partner and Port are not required for a TCP/IP receiver channel.
 - b. The TP Name is not used by TCP/IP channels.
 - c. Ensure that the parameter field values match the values of the sender channel definition of the same name on the remote host.
5. Press PF5 (Add) to add the new channel definition.

Appendix G. MQSeries clients

An MQSeries client is an MQSeries system that does not include a queue manager. The MQSeries client code directs MQI calls from applications running on the client system to a queue manager on an MQSeries server system to which it is connected.

This appendix provides information about MQSeries clients that is specific to MQSeries for VSE/ESA V2.1. It should be used in conjunction with the *MQSeries Clients* book.

Client support

MQSeries for VSE/ESA V2.1 can function as an MQSeries server system to all MQSeries clients that can connect to the server using the TCP/IP protocol. However, there is no MQSeries for VSE/ESA V2.1 client.

All current clients are available from the MQSeries Web site at:

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

When an MQSeries client connects to a queue manager on MQSeries for VSE/ESA V2.1, the client can issue MQI calls as if the queue manager were local to the client program. Valid MQI calls are:

MQCLOSE	Close object
MQCONN	Connect queue manager
MQDISC	Disconnect queue manager
MQGET	Get message
MQINQ	Inquire about object attributes
MQOPEN	Open message queue
MQPUT	Put message
MQPUT1	Put one message
MQSET	Set object attributes

Security considerations

MQSeries for VSE/ESA V2.1 does not support the use of channel security exits for the validation of clients.

After a client has successfully identified and connected to the MQServer program using a valid channel definition, the client can issue any of the supported MQI calls listed above.

Security can be implemented at the client end, provided security exits are supported on the client platform.

Client code-page conversion tables

For client/server code-page conversion, the MQSeries server uses the Language Environment (LE) code set conversion facilities:

genxlt utility

Generates a translation table for use by the iconv utility and the iconv functions to perform code page conversion.

iconv utility

Converts a file from one code set encoding to another.

iconv functions

Functions used by the MQSeries server to perform all code-page conversions.

The `genxlt` utility generates object output that is link-edited to produce a phase. Each phase represents a translation table between two code pages. Consequently, the range of supported client/server code pages is determined by the number of phases generated using the `genxlt` utility.

The `genxlt` and `iconv` language environment utilities are fully documented in the *C Run-Time Programming Guide*, which documents all the supplied code-set converters.

You are **strongly recommended** to read the section on code-page conversion in the *C Run-Time Programming Guide*. If you need code conversion for pages that are not provided, you can edit the appropriate source-code modules and build the converters.

Appendix H. System messages

This appendix describes the messages issued by MQSeries.

MQSeries generates both internal and external messages. Internal messages are generated when an application program activates MQSeries and an abnormal condition occurs.

These messages are stored on the system log queue when it is available; otherwise, the CICS CSMT Transient Data (TD) queue is used.

API system messages

These messages consist of five lines of text, each with a maximum of 78 characters, together with two lines of error code information as follows:

Line 1 –

```
MQInnnnn PRG:ppppppp TRN:tttt TRM:rrrr TSK:cccc mm/dd/yy hh:mm:ss
```

Where:

```
nnnnnn  MQSeries message code – see “MQSeries message codes” on
         page 203
ppppppp  CICS Program name
tttt     CICS Transaction code
rrrr     CICS Term ID
cccc     CICS Task ID
mm/dd/yy Date
hh:mm:ss Time
```

Line 2 – Textual description of message

Line 3 – Queue name, if available

Line 4 – Channel name, if available

Line 5 – Detail of message (optional)

Line 6 –

```
EIBFN:fff EIBRCODE:rrrrrrrrrr EXEC LINE: 11111
```

Where:

```
fff      EIBFN value at time of condition
rrrrrrrr EIBRCODE
11111    The DEBUG CICS command number
```

Line 7 –

```
EIBRESP: rrrrrrrr EIBRESP2: ssssssss EIBRSRCE:ccccccc ABCODE: aaaa
```

Where:

```
rrrrrrrr EIBRESP
ssssssss EIBRESP2
```

```
ccccccc EIBRSRCE  
aaaa    CICS ABENDCODE
```

MQSeries message definitions

Each MQSeries message provides the following information:

Explanation:	This section explains what the message or code means, why it occurred, and what caused it.
Function	This section indicates which modules issued the message, to assist in diagnosing problems.
Severity	Severity values have the following meanings: 0 An information message. No error has occurred. 4 A warning message. A condition has been detected of which you should be aware. You may need to take further action. 8 An error message. An error has been detected that the system typically corrects. However, you may have to intervene. 10 A severe error message. An error has been detected that may severely affect user or system operation. This requires your immediate intervention. 12 A fatal error message. An error has been detected that is so severe that it causes one of the system components to end. This requires your immediate intervention.
Operator action	If an operator response is necessary, this section describes what the appropriate responses are, and what their effect is. If this information is omitted, no operator response is required.
System action	This part describes what is happening as a result of the condition causing the message or code. If this information is omitted, no system action is taken.

MQSeries messages

MQSeries system messages are numbered 000000 through 900000, and they are listed in this book in numeric order. However, not all numbers have been used, therefore, the list is not continuous.

MQSeries console messages are numbered from MQI0001 onwards, and they are listed in this book in numeric order; see “Console Messages” on page 217.

MQSeries message codes

000000 SYSTEM STARTED**Explanation:** System has been Initialized.**Function:** Master Terminal, Sender, Receiver**Severity:** 0**Operator action:** None**System action:** None**000003 CHANNEL MESSAGE SEQUENCE NUMBER ERROR****Explanation:** The received MSN does not match the expected MSN.**Function:** Receiver**Severity:** 8**Operator action:**

1. Review the EXPECT MSN and the RECEIVED MSN in the detail portion of the message.
2. Identify the cause (proper running should preclude this occurrence).
3. Reset the appropriate MSN so that the sender and receiver channel MSNs are equal.
4. Restart communication.

System action: Fatal error – communication is ended.**000004 SYNCH MSG DUP****Explanation:** The received message may be duplicated.**Function:** Receiver**Severity:** 0**Operator action:** None**System action:** Continue negotiating.**000007 LU62 SESSION STARTED****Explanation:** A communication session was established by MQPRECV.**Function:** Receiver**Severity:** 0**Operator action:** None**System action:** None.**000010 LU62 FREE ERROR****Explanation:** For Program MQPRECV, upon completion of a RECEIVE command neither the EIBFREE nor the EIBERR fields has low values.

For Program MQPSEND as a Server, upon completion of a RECEIVE command at least one of EIBERR, EIBRECV and EIBFREE is not equal to a low value.

As a Server or Sender, upon receipt of an acknowledgment of messages sent, the EIBFREE is not equal to a low value and the EIBERR is equal to a low value.

Function: Sender, Receiver**Severity:** 12**Operator action:**

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the EIBTRMID, which is the principal facility associated with this error. Locate any messages associated with this principal facility.

2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.

3. Correct the problem and restart communication.

System action: Fatal error – Communication is ended.**000011 LU62 EIB ERROR****Explanation:**

1. As a Server, upon completion of a RECEIVE, the EIBERR is not equal to a low values.
2. As a Server or Sender, upon receipt of an acknowledgment of messages sent, the EIBERR is not equal to a low value.

Function: Sender, Server**Severity:** 12**Operator action:**

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.

2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.

3. Correct the problem and restart communication.

System action: Fatal error – Communication is ended.**000012 LU62 STAT ERROR****Explanation:** As a Server or Sender, upon receipt of an acknowledgment of messages sent, the EIBRECV is not equal to a low value.**Function:** Sender, Server**Severity:** 12**Operator action:**

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.

2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.

3. Correct the problem and restart communication.

System action: Fatal error – communication is ended.**000013 LU62 ALLOC ERROR****Explanation:** As a Sender, upon completion of an ALLOCATE command, EIBRCODE is not equal to a low value and all retries have been performed.**Function:** Sender**Severity:** 12**Operator action:**

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the

EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.

2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.
3. Correct problem and restart communication.

System action: Fatal error – communication is ended.

000014 LU62 ALLOC RETRY ERROR

Explanation: As a Sender upon completion of an ALLOCATE command, EIBRCODE is not equal to low values and all retry attempts have not been performed.

Function: Sender

Severity: 10

Operator action:

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.
2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.

System action: Severe error – allocation is retried until allocation is successful or the retry count equals zero.

000015 LU62 CONN ERROR

Explanation: As a Sender, upon completion of a CONNECT PROCESS command, EIBRCODE is not equal to low values.

Function: Sender

Severity: 12

Operator action:

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.
2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.
3. Correct the problem and restart communication.

System action: Fatal error – communication is ended.

000016 LU62 SEND ERROR

Explanation: As a Sender or Server, upon completion of a SEND command, EIBRCODE is not equal to a low value.

Function: Sender, Server

Severity: 12

Operator action:

1. Review System log or error TD queue for messages prior to this message. TRM in the error message contains the EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.

2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.

3. Correct the problem and restart communication.

System action: Fatal error – communication is ended.

000017 REMOTE SITE DEALLOCATED CONVERSATION

Explanation: As a Server, if a get request returns a “no message available response”, a RECEIVE command is processed. Any of the following conditions can cause this response:

1. EIBRECV is equal to a low value, or
2. EIBFREE is equal to a low value, or
3. EIBERR is equal to a low value.

Function: Sender

Severity: 8

Operator action: This is an informational message and no additional user action is normally required. The requester, by deallocating the conversation (in response to no messages being available) has caused the server to end communication.

System action: Communication is ended.

000023 INVLD RESP TYPE

Explanation: The SENDER received a response message that does not conform to the format expected.

Function: Sender

Severity: 8

Operator action: Review System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate the receiver and requester processes for programming errors.

System action: Fatal error – Communication is ended.

000024 INVLD RESP MSN

Explanation: (Reserved)

000025 FATAL RESP TYPE

Explanation: (Reserved)

000026 RECOVERABLE RESP TYPE

Explanation: (Reserved)

000029 PARSER MSN ERROR

Explanation: (Reserved)

000030 PARSER TYPE ERROR

Explanation: (Reserved)

000031 PARSER PDM ERROR

Explanation: (Reserved)

000032 PARSER SID ERROR

Explanation: (Reserved)

000033 PARSER PN ERROR

Explanation: (Reserved)

000034 PARSER KEY ERROR**Explanation:** (Reserved)

000035 PARSER APID ERROR**Explanation:** (Reserved)

000038 PARSER ORG DT ERROR**Explanation:** (Reserved)

000039 PARSER ORIG MSN ERROR**Explanation:** (Reserved)

000040 PARSER BODY ERROR**Explanation:** (Reserved)

000041 PARSER STATUS ERROR**Explanation:** The received message does not have the proper status value.**Function:** Sender, Receiver**Severity:** 8**Operator action:** Review System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate the sender process for programming errors.**System action:** Fatal error – communication is ended.

000042 PARSER LENGTH ERROR**Explanation:** The received message does not have the proper length value.**Function:** Sender, Server**Severity:** 8**Operator action:** Review System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate the sender process for programming errors.**System action:** Fatal error – communication is ended.

000051 QUEUE CONNECTION ERROR**Explanation:** The queue manager cannot be connected.**Function:** Sender, Server**Severity:** 12**Operator action:** Review the System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate the sender process for programming errors.**System action:** Fatal error – communication is ended.

000052 QUEUE OPEN ERROR**Explanation:** The Server or Sender could not open the associated transmission queue.**Function:** Sender, Server**Severity:** 12**Operator action:**

- Review the following fields in the error message:
 - QUEUE ID - Transmission queue name that failed.
 - CHANNEL ID - channel name that was connected. This channel identifies the corresponding transmission queue.
 - Last line of error message - Reason code returned from queuer and corresponding description.
- Correct the problem and restart communications.

System action: Fatal error – communication is ended.

000053 QUEUE GET ERROR**Explanation:** The Server or Sender could not get a message from the associated transmission queue even if there are messages in the transmission queue.**Function:** Sender, Server**Severity:** 12**Operator action:**

- Review the following fields in the error message:
 - QUEUE ID - Transmission queue name that failed.
 - CHANNEL ID - channel name that was connected. This channel identifies the corresponding transmission queue.
 - Last line of error message - Reason code returned from queuer and corresponding description.
- Correct the problem and restart communications.

System action: Fatal error – communication is ended.

000054 QUEUE PUT ERROR**Explanation:** The RECEIVER could not put a message to an application queue.**Function:** Receiver**Severity:** 12

- Review the following fields in the error message:
 - QUEUE ID - application queue name that failed.
 - CHANNEL ID - channel name that was connected.
 - Last line of error message - Reason code returned from queuer and corresponding description.
- User action is based upon returned reason code.

For reason code MQRC-Q-FULL (2053) or MQRC-Q-SPACE-NOT-AVAILABLE (2056). The destination application queue was full and the message was placed on the dead letter queue. Determine if the destination queue should be expanded to accommodate more messages or an alternate destination used.

All other reason codes, correct the problem and restart communications.

System action: There are two possible system actions based upon the reason code returned:

- Reason code of MQRC-Q-FULL or MQRC-Q-SPACE-NOT-AVAILABLE. Communication proceeds normally after the first MQPUT call failed to put a message on the dead-letter queue.
- All other reason codes constitute a fatal error and communication is ended.

000055 QUEUE PUT1 ERROR**Explanation:** The RECEIVER could not put a message to the dead-letter queue.**Function:** Receiver**Severity:** 12**Operator action:**

- Review the following fields in the error message:
 - QUEUE ID - Dead-letter queue name that failed.
 - CHANNEL ID - channel name that was connected.

Last line of error message - Reason code returned from queuer and corresponding description.

2. Correct the problem and restart communications.

System action: Fatal error – communication is ended.

000056 QUEUE CLOSE ERROR

Explanation: The RECEIVER could not close an application queue.

Function: Receiver

Severity: 0

Operator action:

1. Review the following fields in the error message:

QUEUE ID - Application queue name that failed.

CHANNEL ID - channel name that was connected.

Last line of error message - Reason code returned from queuer and corresponding description.

2. Correct the problem and restart communications.

System action: Communication proceeds normally, however, the resources that remain open will be closed at an appropriate time.

000057 QUEUE DISC ERROR

Explanation: An error has occurred to DISCONNECT the connecting Queue Manager.

Function: Sender, Receiver

Severity: 12

Operator action: Review the System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate the sender process for programming errors.

System action: Fatal error – communication is ended.

000060 UNDEFINED QUEUE ERROR

Explanation: (Reserved)

000080 RECV RETURN LON STATUS

Explanation: (Reserved)

000081 RECV RETURN LON TYPE

Explanation: (Reserved)

000091 SIDRC RETURN FORMAT

Explanation: (Reserved)

000100 FUNCTION STARTED

Explanation: The requested function has been started

Function: Master Terminal

Severity: 0

Operator action: none

System action: Function is started

001000 FUNCTION DONE

Explanation: The requested function has been completed

Function: Master Terminal

Severity: 0

Operator action: none

System action: Function is completed.

001090 FUNCTION NOT DONE

Explanation: The requested function was ended because of an error. The function was not completed.

Function: Master Terminal

Severity: 0

Operator action: Review the associated message prior to this one.

System action: Function is ended with an error.

005000 CHANNEL CONNECTED

Explanation: Channel connection is successful.

Function: Sender, Receiver

Severity: 0

Operator action: None

System action: Platform negotiation begins.

005001 CHANNEL NEGOTIATIONS ACCEPTED

Explanation: Channel has completed negotiation with the other platform.

Function: Sender, Receiver

Severity: 0

Operator action: None

System action: Message queue is opened.

005002 CHANNEL QUEUE OPENED

Explanation: Channel queue has been opened successfully.

Function: Sender, Receiver

Severity: 0

Operator action: None

System action: Message transfer begins.

005003 CHANNEL LU 6.2 CONNECTED

Explanation: LU 6.2 connection established.

Function: Sender, Receiver

Severity: 0

Operator action: None

System action: LU 6.2 conversation begins.

005004 CHANNEL RECEIVER ALLOCATED

Explanation: (Reserved)

005005 CHANNEL QUEUE EMPTY.

Explanation: (Reserved)

005006 CHANNEL QUEUE CLOSED

Explanation: Channel has successfully closed queue.

Function: Sender, Receiver

Severity: 0

Operator action: None

System action: Channel is disconnected.

005007 CHANNEL DISCONNECTED

Explanation: Channel has been disconnected from the other platform.

Function: Sender, Receiver

Severity: 0

Operator action: None

System action: Channel is shutdown.

005008 CHANNEL SHUTDOWN**Explanation:** Channel has been completely shutdown.**Function:** Sender, Receiver**Severity:** 0**Operator action:** None**System action:** Channel is marked INACTIVE.

005009 CHANNEL SHUTDOWN REQUEST SEND**Explanation:** (Reserved)

006003 CHANNEL TCP/IP CONNECTED**Explanation:** TCP/IP connection established.**Function:** Sender, Receiver**Severity:** 0**Operator action:** None**System action:** TCP/IP conversation will begin.

006007 TCP/IP SESSION STARTED**Explanation:** A communication session was started by the Receiver MCA.**Function:** Receiver**Severity:** 0**Operator action:** None**System action:** None

006010 TCP/IP FREE ERROR**Explanation:** A TCP/IP connection terminated prematurely, possibly due to incomplete data from a remote connection.**Function:** Sender, Receiver**Severity:** 12**Operator action:**

1. Review system log or error TD Queue for messages prior to this message. TRM in the error message contains the EIBTRMID which is the principal facility associated with this error. Locate any messages associated with this principal facility.
2. Correct the problem and restart communication.

System action: Fatal error - communication is ended.

006013 TCP/IP ALLOC ERROR for program MQPTCPV**Explanation:** The MQ/Server program was unable to allocate memory.**Function:** Server**Severity:** 12**Operator action:** This error occurs when there is insufficient memory resource available. Check that other processes are not erroneously allocating memory, or rerun the client/server conversation when the VSE host is less busy.**System action:** Fatal error - Communication is ended.

006015 TCP/IP CONN ERROR for Program MQPSEND**Explanation:** As a Sender upon completion of a CONNECT request, received an invalid return code.**Function:** Sender**Severity:** 12**Operator action:**

1. Review system log or error TD Queue for messages prior to this message.
2. Produce an auxiliary trace to determine the return code returned from the connect.

3. Correct problem and restart communication.

System action: Fatal error - communication is not established.

006016 TCP/IP SEND ERROR for Program MQPSEND**Explanation:** As a Sender or Server upon completion of a SEND request received an invalid return code.**Function:** Sender, Server**Severity:** 12**Operator action:**

1. Review the system log or error TD Queue for messages prior to this message.
2. Produce an auxiliary trace to determine the return code returned from the send.
3. Correct the problem and restart communication.

System action: Fatal error - Communication is ended.

006017 TCP/IP RECV/RESP ERROR**Explanation:** An attempt to receive TCP/IP data failed or the response was other than as expected.**Function:** Sender, Receiver**Severity:** 12**Operator action:**

1. Review the system log or error TD Queue for messages prior to this message.
2. Produce an auxiliary trace to determine the return code returned from the receive.
3. Correct the problem and restart communication.

System action: Fatal error - Communication is ended.

006020 TCP/IP SOCKET ERROR**Explanation:** An error occurred when opening or setting the attributes of a TCP/IP socket.**Function:** Server**Severity:** 12**Operator action:**

1. Check that TCP/IP is installed and running.
2. Check that the TCP/IP phase is cataloged in a library that is concatenated ahead of SCEEbase in the LIBDEF of the CICS startup JCL.

System action: Fatal error - communication cannot start.

006021 TCP/IP COMMS ERROR**Explanation:** A TCP/IP conversation ended due to a transport protocol error, or an the use of a conversation was attempted before it was established.**Function:** Server**Severity:** 12**Operator action:**

1. Check that TCP/IP is running.
2. Attempt to restart the conversation.

System action: Fatal error - communication is ended.

006022 TCP/IP LISTENER BIND ERROR

Explanation: The MQSeries Listener program attempted to bind a port number to a TCP/IP socket and was unsuccessful.

Function: Listener

Severity: 12

Operator action:

1. Check that the listener is not already running.
2. Check that another application is not using the port number configured for the Listener.

System action: Fatal error - Listener cannot start.

006023 TCP/IP LISTENER ACCEPT ERROR

Explanation: The MQSeries listener program attempted to accept a remote conversation and failed.

This is not an error if the remote program ended before the conversation was accepted.

Function: Listener

Severity: 8

Operator action:

1. Check that TCP/IP is running.
2. Restart the Listener.

System action: Error - Listener is ended.

006024 TCP/IP LISTENER ERROR

Explanation: The MQSeries listener program ended due to an unexpected error.

Function: Listener

Severity: 12

Operator action:

1. Check that TCP/IP is running.
2. Restart the listener.

System action: Fatal error - listener is ended.

006025 TCP/IP LISTENER STOPPED

Explanation: The MQSeries Listener program ended normally, or due to an error.

Function: Listener

Severity: 0

Operator action:

1. Check the previous log entries for error messages. If there are no previous error messages the listener ended normally.
2. Restart the listener when appropriate.

System action: Listener program ends.

006026 TCP/IP INVALID CHANNEL TYPE

Explanation: The channel type value in a channel definition is not valid.

Function: Server

Severity: 8

Operator action: Check the channel definition documentation for valid channel type values and update the channel definition appropriately.

System action: Error - Channel cannot be started.

006027 TCP/IP NEGOTIATE FAILED

Explanation: The conversation initial negotiation failed.

Function: Server

Severity: 8

Operator action: Check that the local and remote channel definitions are compatible.

System action: Error - channel cannot be started.

006028 TCP/IP PROTOCOL ERROR

Explanation: The protocol type value in a channel definition is not valid.

Function: Server

Severity: 8

Operator action: Check the channel definition documentation for valid protocol type values and update the channel definition appropriately.

System action: Error - Channel cannot be started.

006029 TCP/IP CONNECT FAILED

Explanation: An attempt to establish a TCP/IP connection failed.

Function: Server

Severity: 12

Operator action: Check the channel definition to ensure that the IP address and port number are valid for the intended remote host.

System action: Fatal error - conversation cannot be started.

006030 TCP/IP UNKNOWN REMOTE CHANNEL ID

Explanation: The channel name used in a remote conversation does not exist on the remote host.

Function: Server

Severity: 8

Operator action: Create the channel definition on the remote host.

System action: Error - communication is ended.

006031 TCP/IP REMOTE QMGR NOT AVAILABLE

Explanation: The queue manager identified in for a remote host is currently unavailable.

Function: Server

Severity: 8

Operator action: Start the remote queue manager on the remote host

System action: Error - communication is ended.

006032 TCP/IP CHANNEL STOPPED BY USER

Explanation: The channel being used in a remote conversation has been disabled by a user.

Function: Server

Severity: 8

Operator action: Restart the channel when appropriate.

System action: Error - communication is ended.

006033 TCP/IP CHANNEL NOT ACTIVE

Explanation: An attempt was made to use a channel that is not currently started.

Function: Server

Severity: 8

Operator action: Start the channel when appropriate.

System action: Error - communication is ended.

006034 TCP/IP CHANNEL STOPPED

Explanation: The channel stopped normally, or due to an error

Function: Server

Severity: 8

Operator action:

1. Check previous log messages for errors. If there are no errors, the channel stopped normally.
2. Restart the channel when appropriate.

System action: Channel activity is ended.

006035 TCP/IP SYNCPOINT FAILED

Explanation: An attempt to perform a CICS SYNCPOINT failed.

Function: Server

Severity: 12

Operator action: Check the system log for previous error messages.

System action: Transaction changes are rolled back.

006036 TCP/IP MESSAGE PUT TO DLQ

Explanation: A message could not be delivered and was written instead to a dead-letter queue.

Function: Server

Severity: 8

Operator action: Examine the dead-letter queue for the undelivered message.

System action: Message written to dead letter queue.

006037 TCP/IP INVALID REMOTE CHANNEL TYPE

Explanation: The channel type value in a channel definition on a remote host is not valid.

Function: Server

Severity: 8

Operator action: Check the channel definition documentation for valid channel type values and update the channel definition on the remote host appropriately.

System action: Error - Channel cannot be started.

006038 TCP/IP XMITQ GET INHIBITED

Explanation: The transmission queue identified by a channel definition has GET INHIBIT enabled.

Function: Server

Severity: 8

Operator action: Disable GET INHIBIT on the transmission queue.

System action: Error - messages cannot be read.

006039 TCP/IP REMOTE CHANNEL UNAVAILABLE

Explanation: The channel identified for a remote conversation is unavailable.

Function: Server

Severity: 8

Operator action: Start the channel on the remote host.

System action: Error - channel cannot be started.

006040 TCP/IP CHANNEL ABEND

Explanation: The channel involved in a current conversation was ended due to an error.

Function: Server

Severity: 12

Operator action: Check the system log for previous error messages.

System action: Fatal error - communication is ended.

006041 TCP/IP LISTENER STARTED

Explanation: The MQSeries Listener program started.

Function: Listener

Severity: 0

Operator action: None

System action: Listener ready to accept remote connections.

006042 TCP/IP SERVER STARTED

Explanation: The MQSeries Server program started.

Function: Server

Severity: 0

Operator action: None

System action: MQSeries Server ready to process client requests.

006043 TCP/IP SERVER STOPPED

Explanation: The MQSeries Server program stopped.

Function: Server

Severity: 4

Operator action: Check previous system log messages to determine whether this is normal or due to an error

System action: MQSeries server-client conversation ended.

006044 TCP/IP BAD SERVER COMMAREA

Explanation: The MQSeries Server program was started with a commarea that is not valid.

Function: Server

Severity: 12

Operator action: Check that the MQSeries Server is not being started by a program other than the MQSeries receiver MCA.

System action: Fatal error - MQSeries server program ended.

006045 TCP/IP SERVER ERROR

Explanation: An error occurred while the MQSeries server was processing a client conversation.

Function: Server

Severity: 8

Operator action: Check the system log for previous error messages and respond to these.

System action: MQSeries server program completed with errors.

006999 TCP/IP UNEXPECTED ERROR

Explanation: An unexpected error has occurred.

Function: Server

Severity: 8

Operator action:

1. Check the system log for previous error messages.
2. Check the QCODE in this error message, which is a numeric code that is meaningful to MQ system support.

System action: Error - Server will probably end.

010000 SYSTEM STARTED W/ ERRORS

Explanation: System being initialized but some queue or channel definitions had errors.

Function: Initialization of system

Severity: 12

Operator action:

1. Review System log or error TD queue for messages prior to this message to identify problem definition.
2. Correct the appropriate definitions.
3. Shut down and reinitialize the system.

System action: Erroneous queues or channels are marked as DISABLED.

010001 SYSTEM STARTED W/ FILE ERRORS

Explanation: System being initialized but some queue files had errors.

Function: Initialization of system

Severity: 12

Operator action:

1. Review System log or error TD queue for messages prior to this message to identify problem definition.
2. Correct the appropriate definitions.
3. Shut down and reinitialize the system.

System action: Erroneous queues are marked DISABLED.

010002 SYSTEM STARTED W/ CHANNEL ERRORS

Explanation: System being initialized but some channel definitions had errors.

Function: Initialization of system

Severity: 12

Operator action:

1. Review System log or error TD queue for messages prior to this message to identify problem definition.
2. Correct the appropriate definitions.
3. Shut down and reinitialize the system.

System action: Erroneous channels are marked DISABLED.

010003 SYSTEM STARTED BUT SYSTEM CHANGED

Explanation: System being initialized but some definitions have been added or deleted while initialization is underway.

Function: Initialization of system

Severity: 8

Operator action: Do not perform configuration changes while the system is being initialized. Shut down and then reinitialize the system.

System action: If definitions were added then some of these definitions may not have been used.

100000 SYSTEM STOPPED

Explanation: System being stopped while application is running.

Function: System Shutdown

Severity: 0

Operator action: All applications and channels should be ended before the system is shutdown.

System action: End request.

100010 SYSTEM ACTIVE

Explanation: System being initialized but it is already active.

Function: Initialization of system

Severity: 0

Operator action: Shut down the system and initialize it again.

System action: System initialization not performed.

100011 SYSTEM STARTED W/ NO QUEUES

Explanation: System being initialized but no queue definitions found.

Function: Initialization of system

Severity: 4

Operator action: Add queue definitions and reinitialize system.

System action: System initialized.

100012 SYSTEM STARTED W/ TOO MANY QUEUES

Explanation: System being initialized but too many queues have been defined.

Function: Initialization of system

Severity: 12

Operator action: Delete some queue definitions and reinitialize the system.

System action: System initialized with some queue definitions.

100013 SYSTEM STARTED W/ TOO MANY CHANNELS

Explanation: System being Initialized but too many channel definitions found.

Function: Initialization of system

Severity: 12

Operator action: Delete some channel definitions and initialize the system.

System action: System initialized with some Channels.

100090 SYSTEM STARTED W/ NO SYSTEM DEFINITION

Explanation: System being Initialized but no System Definition found.

Function: Initialization of system

Severity: 12

Operator action: Define Global System Definition and then initialize the system.

System action: System initialization ended.

101000 QUEUE QDEPTH EXCEEDED

Explanation: The queue QDEPTH would have been exceeded if the PUT request had been performed.

Function: General (I/O modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action: Perform one of the following:

1. Empty this queue either through an application or the queue maintenance facility.
2. Expand the QDEPTH number in the QUEUE definition and refresh the information for this queue.

System action: The PUT request was ended and the problem queue marked as "MAX".

101010 QUEUE CONCURRENT UPDATE HAS OCCURRED

Explanation: Two or more update requests were being received at one time for the same QSN record.

Function: General (I/O modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action:

1. Review all ended requests.
2. Reprocess any legitimate requests.

System action: The first request was served and the remaining were rejected.

101015 QUEUE NOT FOUND

Explanation: MQPSSQ, a subroutine to start and stop a queue, reported that the queue to be processed was not defined in the system.

Function: Start and stop queue

Severity: 8

Operator action: Reprocess any ended requests.

System action: The request was ended unsuccessfully.

101090 QUEUE STOPPED

Explanation: A request has been processed against a STOPPED queue.

Function: Start and stop queue

Severity: 4

Operator action: START the problem queue

System action: End the request.

101091 QUEUE DISABLED

Explanation: Queue had errors during initialization.

Function: Initialization of system

Severity: 8

Operator action:

1. Examine the queue definition and file allocation for problems.
2. Reinitialize System.

System action: The problem queue is marked STOPPED.

102090 QUEUE QSN NUMBER LIMIT HAS BEEN REACHED

Explanation: MQPQUE1, a subroutine serving all input-output requests for queues, detected that QSN would exceed the full word limitation of 99,999,999.

Function: General (Input-output modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action: Perform one of the following:

1. Perform file maintenance on this problem queue, for example, running the batch job MQPREORG.
2. Process the on line queue maintenance facility to delete messages using "Delete by Date/time".

System action: The PUT request for this queue was rejected.

102091 QUEUE NO SPACE AVAILABLE FOR PUT

Explanation: Queue encountered a NOSPACE condition for a PUT request.

Function: General (Input-output modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action: Perform one of the following:

1. Perform file maintenance on this problem queue, for example, running the batch job MQPREORG.
2. Process the on line queue maintenance facility to delete messages using "Delete by Date/time".

System action: End the request and mark queue "FULL".

102092 QUEUE NO SPACE AVAILABLE

Explanation: Queue encountered errors for an UPDATE request, NOSPACE condition occurred.

Function: General (Input-output modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action: Perform one of the following:

1. Perform file maintenance on this problem queue, for example, running the batch job MQPREORG.
2. Process the on line queue maintenance facility to delete messages using "Delete by Date/time".

System action: End the request and mark queue "FULL".

104021 DUAL QUEUE ERROR

Explanation: Dual destination queue has been STOPPED or was not initialized properly.

Function: General (Input-output modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action: Perform one of the following:

1. Try to restart the dual queue.
2. Examine and correct the queue and file definition for this queue. Refresh the queue or reinitialize system.

System action: Marked dual queue as "recovery needed".

104022 DUAL QUEUE FILE ERROR

Explanation: Dual destination queue had file error or was not initialized properly.

Function: General (Input-output modules MQPQUE1 and MQPQUE2)

Severity: 8

Operator action: Perform one of the following:

1. Try to restart the dual queue.
2. Examine and correct the queue and file definition for this queue. Refresh the queue or reinitialize system.

System action: Marked dual queue as "recovery needed".

104023 DUAL QUEUE LOGIC ERROR

Explanation: Dual destination queue does not match Source queue.

Function: Master Terminal

Severity: 8

Operator action: Examine and correct the queue and file definition for this queue. Refresh the queue or reinitialize system.

System action: Marked dual queue as "recovery needed".

105090 QUEUE TRIGGER ERROR

Explanation: MQPSSQ, a subroutine to start and stop a queue, encountered an error to start MQ02, a transaction that handles the trigger function.

Function: Start and stop queue

Severity: 12

Operator action: Examine the CICS tables to correct the problem.

System action: The request was ended unsuccessfully.

105091 QUEUE TRIGGER DATA ERROR

Explanation: MQPAIP2, a program handling trigger function, received erroneous data and could not fulfil the request.

Function: Application Interface

Severity: 12

Operator action: Contact system support for MQSeries for VSE.

System action: The request was ended unsuccessfully.

109000 ACTION NOT AUTHORIZED

Explanation: NOAUTH condition flagged by CICS when a resource security check failed.

Function: General (CICS Interface)

Severity: 12

Operator action: Review security mechanism.

System action: The request was ended unsuccessfully.

300000 ACTION NOT SUPPORTED

Explanation: Module has been LINKed with an incorrect function.

Function: General (CICS Interface)

Severity: 12

Operator action: Review application for call format.

System action: End the request.

300010 PROGRAM STARTED INCORRECTLY

Explanation: Module has been STARTed with incorrect function.

Function: General (CICS Interface)

Severity: 12

Operator action: Review application for call format.

System action: End the request.

300020 PROGRAM HAS REPEATED ERRORS

Explanation: MAPFAIL condition raised in Master Terminal (MQMT) panels.

Function: General (CICS Interface)

Severity: 12

Operator action: Review PPT for MAP modules and correct the problem.

System action: End the request.

300030 QUEUE LOCK TABLE IS FULL

Explanation: Insufficient queue lock entries present to insert a new entry.

Function: General (Control Module MQPLOCK)

Severity: 12

Operator action: Review application for multiple message retrieval without a SYNCPOINT. If there is not an application problem, increase the queue lock count to a higher value. Note that this value is used to calculate an incore table.

System action: End the request.

301000 EXPECTED RECORD IS MISSING

Explanation: An expected message was found missing. This normally occurs during a Deletion request.

Function: Master Terminal

Severity: 8

Operator action: Restart the application.

System action: End the request.

301010 DUPLICATE RECORD HAS OCCURRED

Explanation: A duplicate message was found. This normally occurs during a PUT request.

Function: General (MQPQUE1)

Severity: 8

Operator action: Restart the application.

System action: End the request.

309010 QUEUE CHECKPOINT RECORD MISSING

Explanation: A queue checkpoint was requested but no checkpoint record was found on this queue.

Function: Master Terminal

Severity: 12

Operator action: Initialize the system and restart the application.

System action: End the request.

400000 LINK ERROR

Explanation: Unable to perform a LINK request.

Function: General (CICS Interface)

Severity: 12

Operator action: Examine any prior messages for actual problem.

System action: End the request.

400001 LINK DFHCOMMAREA SIZE INCORRECT

Explanation: Expected DFHCOMMAREA length was incorrect.

Function: General (CICS Interface)

Severity: 12

Operator action: Examine any prior messages for the actual problem.

System action: End the request.

400002 LINK DFHCOMMAREA DATA INCORRECT

Explanation: Expected DFHCOMMAREA data is incorrect.

Function: General (CICS Interface)

Severity: 12

Operator action: Examine any prior messages for actual problem.

System action: End the request.

400003 RETURN FROM LINK ERROR

Explanation: A LINK request ended in an abnormal condition.

Function: General (CICS Interface)

Severity: 12

Operator action: Examine any prior messages for actual problem.

System action: End the request.

400010 MOVE ERROR**Explanation:** Internal MOVE of data has found corrupt data.**Function:** General**Severity:** 12**Operator action:** Examine any prior messages for actual problem.**System action:** End the request.

402000 INTERNAL STRUCTURE MISSING**Explanation:** Internal Structure was found missing.**Function:** General**Severity:** 12**Operator action:** Examine any prior messages for actual problem.**System action:** End the request.

402090 INTERNAL STRUCTURE HAS ERRORS**Explanation:** Internal Structure corrupted.**Function:** General**Severity:** 12**Operator action:** Examine any prior messages for the actual problem.**System action:** End the request.

501001 CHANNEL FREE ERROR**Explanation:** (Reserved)

501002 EIB ERROR**Explanation:** RECEIVER encounters an error:

1. Upon completion of a GETMAIN command, the EIBRCODE was not equal to low values, or
2. Upon completion of a RECEIVE command:
 - RESP not equal to TERMERR, and
 - EIBFREE equal to low values, and
 - EIBERR not equal to low values.

Function: Receiver**Severity:** 12**Operator action:**

1. Review the system log or error TD queue for messages prior to this message. TRM in the error message contained the EIBTRMID, which is the principal facility associated with this error. Locate any messages associated with this principal facility.
2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.
3. Correct the problem and restart communications.

System action: Fatal error – communication was ended.

501003 CHANNEL STAT ERROR**Explanation:** (Reserved)

501004 CHANNEL ALLOC ERROR**Explanation:** (Reserved)

501005 CHANNEL ALLOC RETRY ERROR**Explanation:** (Reserved)

501006 CHANNEL CONNECT ERROR**Explanation:** RECEIVER or SENDER cannot connect a channel.**Function:** Sender, Receiver**Severity:** 12**Operator action:**

1. Review the following fields in the error message:

CHANNEL ID - channel name that was being connected.

Last line of error message - Reason code returned from queue and corresponding description.

2. Correct problem and restart communication.

System action: Fatal error – communication was ended.

501008 CHANNEL SEND ERROR**Explanation:** RECEIVER issued a SEND command and its EIBRCODE is not normal (zeros).**Function:** Receiver**Severity:** 12**Operator action:**

1. Review the system log or error TD queue for messages prior to this message. TRM in the error message contained the EIBTRMID, which is the principal facility associated with this error. Locate any messages associated with this principal facility.
2. Review the EIBRCODE, EIBRESP, and EIBRESP2 fields in the detail portion of the message. They contain information about the cause of the problem. Refer to the CICS/ESA Application Programming Reference manual for an explanation of these values.
3. Correct the problem and restart communications.

System action: Fatal error – communication was ended.

501009 RECEIVER RESPONSES WITH ERROR**Explanation:** SENDER received a rejection from RECEIVER to end communication.**Function:** Sender**Severity:** 12**Operator action:** Review the error reason code to determine the reason of the rejection and restart the communication after correction.**System action:** Fatal error – communication was ended.

501010 INVALID RESPONSE TYPE**Explanation:** Unsupported Message Segment Type received.**Function:** Sender**Severity:** 12**Operator action:** Review the Segment type and restart communications without the problem type.**System action:** Fatal error – communication was ended.

501011 CHANNEL RESPONSE MSN ERROR**Explanation:** (Reserved)

501012 CHANNEL RESPONSE HAS FATAL ERROR**Explanation:** (Reserved)**501013 CHANNEL RE-NEGOTIATION****Explanation:** RECEIVER rejects a channel parameter and makes a counter proposal for renegotiation**Function:** Receiver**Severity:** 4**Operator action:** No action is needed unless remote platform can not accept the conflicting parameter. If this happens then the conflicting parameter must be changed on this or the remote platform.**System action:** Reject this proposal and continue on negotiation.**501014 UNKNOWN ENCODING****Explanation:** Transmission Segment Header contains unknown encoding.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Disregard the error and continue on initiation.**501015 INVALID TRANSMISSION SEGMENT HEADER****Explanation:** Transmission Segment Header contains either wrong type or invalid format.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Disregard the error and continue on initiation.**501016 UNSUPPORTED CODED CHARACTER SET ID (CCSID)****Explanation:** Coded character set ID in used is not supported.**Function:** Sender, Receiver**Severity:** 4**Operator action:** none**System action:** Disregard the error and try another CCSID if any.**501017 INVALID MESSAGE SEGMENT HEADER****Explanation:** Message Segment Header is invalid.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Disregard the error and re-try.**501018 INVALID TRANSMISSION QUEUE HEADER****Explanation:** Transmission queue header is invalid.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Disregard the error and re-try.**501019 INITIATION ERROR****Explanation:** Error encountered during initiation.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Disregard the error and continue initiation.**501020 INVALID FAP LEVEL****Explanation:** The protocol in use was not supported.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Disregard the error and continue initiation.**501021 MESSAGE SIZE TOO BIG****Explanation:** The message size was too large to be processed.**Function:** Receiver**Severity:** 4**Operator action:** None**System action:** Select a smaller message size and continue negotiation.**501022 MESSAGE WRAP ERROR****Explanation:** The message sequence number wrap around value could not be accepted.**Function:** Sender, Receiver**Severity:** 4**Operator action:** None**System action:** Select a smaller value and continue negotiation.**501023 QUEUE MANGER DOWN DURING ACCESSING DLQ****Explanation:** The message was not put into the Dead Letter Queue because the system was not running.**Function:** Receiver**Severity:** 8**Operator action:** Initiate the system using MQIT or, through the MQMT panels.**System action:** Process was ended.**501024 QUEUE MANAGER IS DOWN****Explanation:** Communication could not be established because the queue manager is not running.**Function:** Sender, Receiver**Severity:** 8**Operator action:** Initiate the system using MQIT or, through the MQMT panels.**System action:** Process is terminated.**501025 UNKNOWN CHANNEL ID (INBOUND)****Explanation:** The communication cannot be established because the channel id received from the remote system is not defined locally.**Function:** Sender, Receiver**Severity:** 8**Operator action:** Check the channel id to see if it is correct. Define this in the local definitions or correct the remote system as necessary.**System action:** The communication session is terminated.**501026 CHANNEL ERROR****Explanation:** (Reserved)

501027 CHANNEL BUSY

Explanation: SENDER reported there was a message waiting to be queued on the channel name.

Function: Sender

Severity: 12

Operator action:

1. Review the CHANNEL ID field in the error message.
2. Determine why a second channel was started.
3. Validate channel configuration.

System action: Fatal error – communication was ended.

501028 CHANNEL RE-SYNC ERROR

Explanation: Expected TCF-Confirm-Request flag was not turned on in the received initiation message.

Function: Sender, Receiver

Severity: 4

Operator action: none

System action: Disregard the error and continue on initiation.

501029 CHANNEL STATUS ERROR

Explanation: (RESERVED)

501030 MESSAGE LENGTH ERROR

Explanation: RECEIVER encountered one of the following:

1. The length of the application portion of the message specified in the header exceeded the maximum length defined for this channel.
2. The length of the application portion of the message received was not equal to the length specified in the header.

Function: Receiver

Severity: 8

Operator action: For the first explanation:

1. Review the Max Transmission Size and the Max Message Size in the detail portion of the message.
2. Check the configuration of the Receiver channel to ensure that the maximum message size was set correctly.
3. Check the configuration of the Sender.
4. Reconfigure if necessary and restart communication.

For the second explanation:

1. Review the Max Transmission Size and the Max Message Size in the detail portion of the message.
2. Proper running should preclude this occurrence. Investigate the sender and server process for programming errors.
3. Correct the problem and restart communications.

System action: Fatal error – communication was ended.

501031 MESSAGE-PER-BATCH TOO BIG

Explanation: The maximum number of messages allowed in a batch was too large to be handled.

Function: Sender, Receiver

Severity: 4

Operator action: None

System action: Select a smaller size and continue negotiation.

501032 MAX TRANSMISSION SIZE TOO BIG

Explanation: The maximum transmission size was too large to be handled.

Function: Sender, Receiver

Severity: 4

Operator action: None

System action: Select a smaller size and continue negotiation.

501050 RESET MSN

Explanation: Remote platform Message Sequence Number was reset.

Function: Sender, Receiver

Severity: 4

Operator action: None

System action: Validate that the MSN was within one of the current MSN on this platform.

All messages starting with 6000 are severe messages displayed on the CICS terminals from which MQSeries Administrator Dialogs (MQMT) have been started.

These messages indicate failures in the MQSeries code itself. Each message number is followed by the program name in which the failure occurred.

If after checking, and making any possible corrections, the problem persists, report this to your IBM support organization. You should include the message number, together with the names of the modules and any other information.

600001 Prog: xxxxxxxx Error detected. Contact Support.

Explanation: CICS has detected an error condition not handled by a specific routine.

Severity: 8

Operator action: Report to IBM

System action: The dialog was ended.

600005 Prog: xxxxxxxx ABEND Code zzzz Contact Support.

Explanation: The program ended due to a CICS problem and the ABEND code zzzz was returned to a HANDLE ABEND routine.

Severity: 8

Operator action: Report to IBM

System action: The dialog was ended.

600007 Prog: xxxxxxxx File: yyyyyyy Not Found. Contact Support.

Explanation: A request has been issued against the file yyyyyyy, but it was not defined in the FCT

Severity: 8

Operator action: Contact your system administrator and check whether all MQSeries files were defined in the CICS File Control Table (FCT), and physically allocated by VSAM.

System action: The dialog was ended.

600009 Prog: xxxxxxxx File: yyyyyyy DISABLED. Contact Support.

Explanation: CICS tried to access the file yyyyyyy which was not enabled.

Severity: 8

Operator action: Use "CEMT S DATA" to set the file ENABLED. If the DISABLED status persists, check with the System Administrator.

System action: The dialog was ended.

600011 Prog: xxxxxxxx File: yyyyyyy ILLOGIC error. Contact Support.

Explanation: Usually this is related to file input and output. This condition is returned by CICS when the error does not fall within one of the other CICS response categories.

Severity: 8

Operator action: Report to IBM

System action: The dialog was ended.

600017 Prog: xxxxxxxx File: yyyyyyy I/O error. Contact Support.

Explanation: Normally this is due to hardware errors.

Severity: 8

Operator action: Check the System console for more details.

System action: The dialog was ended.

600019 Prog: xxxxxxxx File: yyyyyyy Record not found. Contact Support.

Explanation: The program tried to read a record but the request failed.

Severity: 8

Operator action: Report to IBM.

System action: The dialog was ended.

600021 Prog: xxxxxxxx File: yyyyyyy is not open. Contact Support.

Explanation: CICS tried to access a file which had not been opened, and was unable to open it. This can happen when the file is already in use by another partition.

Severity: 8

Operator action: Use "CEMT I DATA" and try to open it manually.

System action: The dialog was ended.

600023 Prog: xxxxxxxx INVREQ error Contact Support.

Explanation: A request was received by CICS and cannot be processed for various reasons.

Severity: 8

Operator action: Report to IBM

System action: The dialog was ended.

600025 Prog: xxxxxxxx MAPFAIL error Contact Support.

Explanation: CICS was unable to display a BMS map on the terminal.

Severity: 8

Operator action: Report to IBM

System action: The dialog was ended.

600027 Prog: xxxxxxxx TRANSID error Contact Support.

Explanation: MQSeries tried to initiate a transaction, but this transaction was not found in the CICS tables.

Severity: 8

Operator action: This is probably an installation error. Check whether the MQSeries group has been correctly installed in the DFHCSD file, and activated. Use CEMT I TRAN(MQ*) to verify this. If everything appears to be correct, report the problem to IBM.

System action: The dialog was ended.

800000 CICS ERROR CONDITION REACHED

Explanation: ERROR condition of CICS occurred.

Function: General (CICS Interface)

Severity: 12

Operator action: Investigate the error.

System action: End the request.

800010 INVALID REQUEST CONDITION

Explanation: INVREQ (Invalid Request) condition of CICS reached.

Function: General (CICS Interface)

Severity: 12

Operator action: Investigate the error.

System action: End the request.

800011 ILLOGIC CONDITION

Explanation: ILLOGIC condition of CICS occurred.

Function: General (CICS Interface)

Severity: 12

Operator action: Investigate the error.

System action: End the request.

800090 ERROR CONDITION DURING CHECKPOINT PROCESSING

Explanation: A general error occurred while processing the checkpoint record of a queue file.

Function: General (I/O modules MQPQUE1 and MQPQUE2)

Severity: 12

Operator action: Use LISTCAT to review the VSAM file containing this queue file.

System action: End the request.

800099 CICS ABEND CONDITION REACHED

Explanation: ABEND condition of CICS occurred.

Function: General (CICS Interface)

Severity: 12

Operator action: Investigate the error.

System action: End the request.

801012 FILE NOTOPEN CONDITION**Explanation:** A CICS file entry has been CLOSED.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install of CICS table.**System action:** End the request.**801019 DISABLE CONDITION****Explanation:** A CICS table entry has been DISABLED.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install of CICS table.**System action:** End the request.**802000 NO STORAGE CONDITION****Explanation:** A CICS storage is not available.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check that the user task has not freed storage.**System action:** End the request.**803001 LENGTH ERROR CONDITION****Explanation:** A record was larger than expected.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check that the product has been correctly installed.**System action:** End the request.**808000 MAPFAIL CONDITION****Explanation:** A CICS transaction is missing.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install of CICS PPT table for maps.**System action:** End the request.**809000 PGMIDERR CONDITION****Explanation:** A CICS program id is missing.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install of CICS PPT table.**System action:** End the request.**809010 FILEID CONDITION****Explanation:** No file was available to process.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install for CICS FCT table.**System action:** End the request.**809011 NOFILE CONDITION****Explanation:** No file was available to process.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install for CICS FCT table.**System action:** End the request.**809012 IO ERROR CONDITION****Explanation:** A CICS I/O error has occurred.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check CICS log and EIB codes.**System action:** End the request.**809050 TRANIDERR CONDITION****Explanation:** A CICS transaction is missing.**Function:** General (CICS Interface)**Severity:** 12**Operator action:** Check install of CICS PCT table.**System action:** End the request.**900000 ENVIRONMENT RECORD****Explanation:** Setup of Environment has not been performed.**Function:** Set up system**Severity:** 8**Operator action:** Process Transaction MQSE to setup Environment.**System action:** End the request.

Console Messages

The following messages are informational only.

MQI0001I MQSeries for VSE/ESA starting initialization.**MQI0003I MQSeries initialization is complete.****MQI0005I FILE: QUEUE: (nnnnnn)****Explanation:** nnnnnn can be one of:

Not found

Cannot enable

Cannot open

MQI0011I MQSeries for VSE/ESA ending.**MQI0013I MQSeries termination is complete.****MQI0021I MQSeries for VSE/ESA environment initializing.****MQI0023I MQSeries for VSE/ESA environment complete.****MQI0025I MQSeries for VSE/ESA shutdown complete.**

Appendix I. Notices

This information was developed for products and services offered in the United States. IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM documentation or non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those documents or Web sites. The materials for those documents or Web sites are not part of the materials for this IBM product and use of those documents or Web sites is at your own risk.

Notices

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.

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

ACF/VTAM	AIX	AS/400
BookManager	CICS	CICS/VSE
IBM	MQSeries	MVS/ESA
OS/2	OS/400	System/370
System/390		

Lotus Notes is a registered trademark of Lotus Development Corporation in the United States, or other countries, or both.

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

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

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

Glossary of terms and abbreviations

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

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

A

abend reason code. A 4-byte hexadecimal code that uniquely identifies a problem with MQSeries for MVS/ESA. A complete list of MQSeries for MVS/ESA abend reason codes and their explanations is contained in the *MQSeries for MVS/ESA Messages and Codes* manual.

active log. See *recovery log*.

adapter. An interface between MQSeries for MVS/ESA and TSO, IMS, CICS, or batch address spaces. An adapter is an attachment facility that enables applications to access MQSeries services.

address space. The area of virtual storage available for a particular job.

address space identifier (ASID). A unique, system-assigned identifier for an address space.

alert. A message sent to a management services focal point in a network to identify a problem or an impending problem.

alert monitor. In MQSeries for MVS/ESA, a component of the CICS adapter that handles unscheduled events occurring as a result of connection requests to MQSeries for MVS/ESA.

alias queue object. An MQSeries object, the name of which is an alias for a base queue defined to the local queue manager. When an application or a queue manager uses an alias queue, the alias name is resolved and the requested operation is performed on the associated base queue.

allied address space. See *ally*.

ally. An MVS address space that is connected to MQSeries for MVS/ESA.

APAR. Authorized program analysis report.

application environment. The software facilities that are accessible by an application program. On the MVS platform, CICS and IMS are examples of application environments.

application queue. A queue used by an application.

ASID. Address space identifier.

asynchronous messaging. A method of communication between programs in which programs place messages on message queues. With asynchronous messaging, the sending program proceeds with its own processing without waiting for a reply to its message. Contrast with *synchronous messaging*.

attribute. One of a set of properties that defines the characteristics of an MQSeries object.

authorization checks. Security checks that are performed when a user tries to open an MQSeries object.

authorized program analysis report (APAR). A report of a problem caused by a suspected defect in a current, unaltered release of a program.

B

backout. An operation that reverses all the changes made during the current unit of recovery or unit of work. After the operation is complete, a new unit of recovery or unit of work begins. Contrast with *commit*.

basic mapping support (BMS). An interface between CICS and application programs that formats input and output display data and routes multiple-page output messages without regard for control characters used by various terminals.

BMS. Basic mapping support.

browse. In message queuing, to use the MQGET call to copy a message without removing it from the queue. See also *get*.

browse cursor. In message queuing, an indicator used when browsing a queue to identify the message that is next in sequence.

buffer pool • distributed application

buffer pool. An area of main storage used for MQSeries for MVS/ESA queues, messages, and object definitions. See also *page set*.

C

call back. In MQSeries, a requester message channel initiates a transfer from a sender channel by first calling the sender, then closing down and awaiting a call back.

CCSID. Coded character set identifier.

CDF. Channel definition file.

channel. See *message channel*.

channel control function (CCF). In MQSeries, a program to move messages from a transmission queue to a communication link, and from a communication link to a local queue, together with an operator panel interface to allow the setup and control of channels.

channel definition file (CDF). In MQSeries, a file containing communication channel definitions that associate transmission queues with communication links.

channel event. An event indicating that a channel instance has become available or unavailable. Channel events are generated on the queue managers at both ends of the channel.

checkpoint. A time when significant information is written on the log. Contrast with *syncpoint*.

CI. Control interval.

client. A run-time component that provides access to queuing services on a server for local user applications. The queues used by the applications reside on the server. See also *MQSeries client*.

client application. An application, running on a workstation and linked to a client, that gives the application access to queuing services on a server.

client connection channel type. The type of MQI channel definition associated with an MQSeries client. See also *server connection channel type*.

coded character set identifier (CCSID). The name of a coded set of characters and their code point assignments.

command. In MQSeries, an instruction that can be carried out by the queue manager.

command processor. The MQSeries component that processes commands.

command server. The MQSeries component that reads commands from the system-command input queue, verifies them, and passes valid commands to the command processor.

commit. An operation that applies all the changes made during the current unit of recovery or unit of work. After the operation is complete, a new unit of recovery or unit of work begins. Contrast with *backout*.

completion code. A return code indicating how an MQI call has ended.

connect. To provide a queue manager connection handle, which an application uses on subsequent MQI calls. The connection is made either by the MQCONN call, or automatically by the MQOPEN call.

connection handle. The identifier or token by which a program accesses the queue manager to which it is connected.

context. Information about the origin of a message.

control interval (CI). A fixed-length area of direct access storage in which VSAM stores records and creates distributed free spaces. The control interval is the unit of information that VSAM transmits to or from direct access storage.

controlled shutdown. See *quiesced shutdown*.

CPF. Command prefix.

D

DAE. Dump analysis and elimination.

datagram. The simplest message that MQSeries supports. This type of message does not require a reply.

DCI. Data conversion interface.

dead-letter queue (DLQ). A queue to which a queue manager or application sends messages that it cannot deliver to their correct destination.

default object. A definition of an object (for example, a queue) with all attributes defined.

deferred connection. A pending event that is activated when a CICS subsystem tries to connect to MQSeries for MVS/ESA before MQSeries for MVS/ESA has been started.

distributed application. In message queuing, a set of application programs that can each be connected to a different queue manager, but that collectively constitute a single application.

distributed queue management (DQM). In message queuing, the setup and control of message channels to queue managers on other systems.

DLQ. Dead-letter queue.

DQM. Distributed queue management.

dump analysis and elimination (DAE). An MVS service that enables an installation to suppress SVC dumps and ABEND SYSUDUMP dumps that are not needed because they duplicate previously written dumps.

E

environment. See *application environment*.

ESM. External security manager.

event. See *channel event*.

event log. See *application log*.

external security manager (ESM). A security product that is invoked by the MVS System Authorization Facility.

F

FIFO. First-in-first-out.

first-in-first-out (FIFO). A queuing technique in which the next item to be retrieved is the item that has been in the queue for the longest time. (A)

forced shutdown. A type of shutdown of the CICS adapter where the adapter immediately disconnects from MQSeries for MVS/ESA, regardless of the state of any currently active tasks. Contrast with *quiesced shutdown*.

FRR. Functional recovery routine.

functional recovery routine (FRR). An MVS recovery/termination manager facility that enables a recovery routine to gain control in the event of a program interrupt.

G

get. In message queuing, to use the MQGET call to remove a message from a queue. See also *browse*.

H

handle. See *connection handle* and *object handle*.

I

immediate shutdown. In MQSeries, a shutdown of a queue manager that does not wait for applications to disconnect. Current MQI calls are allowed to complete, but new MQI calls fail after an immediate shutdown has been requested. Contrast with *quiesced shutdown* and *preemptive shutdown*.

initiation queue. A local queue on which the queue manager puts trigger messages.

input/output parameter. A parameter of an MQI call in which you supply information when you make the call, and in which the queue manager changes the information when the call completes or fails.

input parameter. A parameter of an MQI call in which you supply information when you make the call.

Interactive System Productivity Facility (ISPF). An IBM licensed program that serves as a full-screen editor and dialog manager. It is used for writing application programs, and provides a means of generating standard screen panels and interactive dialogues between the application programmer and terminal user.

IP Address.. Internet Protocol address. A 4-part decimal value that uniquely identifies a remote host, for example, 1.20.33.444

L

listener. In MQSeries distributed queuing, a program that monitors for incoming network connections. On MQSeries for VSE/ESA V2.1, the listener waits exclusively for TCP/IP connection requests and starts the Receiver MCA.

local definition. An MQSeries object belonging to a local queue manager.

local definition of a remote queue. An MQSeries object belonging to a local queue manager. This object defines the attributes of a queue that is owned by another queue manager. In addition, it is used for queue-manager aliasing and reply-to-queue aliasing.

local queue. A queue that belongs to the local queue manager. A local queue can contain a list of messages waiting to be processed. Contrast with *remote queue*.

local queue manager. The queue manager to which a program is connected and that provides message queuing services to the program. Queue managers to which a program is not connected are called *remote queue managers*, even if they are running on the same system as the program.

log • persistent message

log. In MQSeries, a file recording the work done by queue managers while they receive, transmit, and deliver messages.

logical unit of work (LUW). See *unit of work*.

M

machine check interrupt. An interruption that occurs as a result of an equipment malfunction or error. A machine check interrupt can be either hardware recoverable, software recoverable, or nonrecoverable.

MCA. Message channel agent.

MCI. Message channel interface.

message. (1) In message queuing applications, a communication sent between programs. See also *persistent message* and *nonpersistent message*. (2) In system programming, information intended for the terminal operator or system administrator.

message channel. In distributed message queuing, a mechanism for moving messages from one queue manager to another. A message channel comprises two message channel agents (a sender and a receiver) and a communication link. Contrast with *MQI channel*.

message channel agent (MCA). A program that transmits prepared messages from a transmission queue to a communication link, or from a communication link to a destination queue.

message descriptor. Control information describing the message format and presentation that is carried as part of an MQSeries message. The format of the message descriptor is defined by the MQMD structure.

message priority. In MQSeries, an attribute of a message that can affect the order in which messages on a queue are retrieved, and whether a trigger event is generated.

message queue. Synonym for *queue*.

message queue interface (MQI). The programming interface provided by the MQSeries queue managers. This programming interface allows application programs to access message queuing services.

message queuing. A programming technique in which each program within an application communicates with the other programs by putting messages on queues.

message sequence numbering. A programming technique in which messages are given unique numbers during transmission over a communication link. This enables the receiving process to check whether all

messages are received, to place them in a queue in the original order, and to discard duplicate messages.

messaging. See *synchronous messaging* and *asynchronous messaging*.

MQI. Message queue interface.

MQI channel. Connects an MQSeries client to a queue manager on a server system, and transfers only MQI calls and responses in a bidirectional manner. Contrast with *message channel*.

MQSeries. A family of IBM licensed programs that provides message queuing services.

MQSeries client. Part of an MQSeries product that can be installed on a system without installing the full queue manager. The MQSeries client accepts MQI calls from applications and communicates with a queue manager on a server system.

N

null character. The character that is represented by X'00'.

O

object. In MQSeries, an object is a queue manager, a queue, a process definition, a channel, a namelist (MVS/ESA only), or a storage class (MVS/ESA only).

object descriptor. A data structure that identifies a particular MQSeries object. Included in the descriptor are the name of the object and the object type.

object handle. The identifier or token by which a program accesses the MQSeries object with which it is working.

output parameter. A parameter of an MQI call in which the queue manager returns information when the call completes or fails.

P

page set. A VSAM data set used when MQSeries for MVS/ESA moves data (for example, queues and messages) from buffers in main storage to permanent backing storage (DASD).

pending event. An unscheduled event that occurs as a result of a connect request from a CICS adapter.

persistent message. A message that survives a restart of the queue manager.

ping. In distributed queuing, a diagnostic aid that uses the exchange of a test message to confirm that a message channel or a TCP/IP connection is functioning.

platform. In MQSeries, the operating system under which a queue manager is running.

point of recovery. In MQSeries for MVS/ESA, the term used to describe a set of backup copies of MQSeries for MVS/ESA page sets and the corresponding log data sets required to recover these page sets. These backup copies provide a potential restart point in the event of page set loss (for example, page set I/O error).

preemptive shutdown. In MQSeries, a shutdown of a queue manager that does not wait for connected applications to disconnect, nor for current MQI calls to complete. Contrast with *immediate shutdown* and *quiesced shutdown*.

Port.. A unique communications identifier used by TCP/IP programs to establish a conversation with a specific application. The target application binds a TCP/IP socket to the unique port number and waits for connection requests for the port from remote hosts.

program temporary fix (PTF). A solution or by-pass of a problem diagnosed by IBM field engineering as the result of a defect in a current, unaltered release of a program.

PTF. Program temporary fix.

Q

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

queue manager. (1) A system program that provides queuing services to applications. It provides an application programming interface so that programs can access messages on the queues that the queue manager owns. See also *local queue manager* and *remote queue manager*. (2) An MQSeries object that defines the attributes of a particular queue manager.

queue manager event. An event that indicates:

- An error condition has occurred in relation to the resources used by a queue manager. For example, a queue is unavailable.
- A significant change has occurred in the queue manager. For example, a queue manager has stopped or started.

queuing. See *message queuing*.

quiesced shutdown. (1) In MQSeries, a shutdown of a queue manager that allows all connected applications to disconnect. Contrast with *immediate shutdown* and *preemptive shutdown*. (2) A type of shutdown of the CICS adapter where the adapter disconnects from MQSeries, but only after all the currently active tasks have been completed. Contrast with *forced shutdown*.

quiescing. In MQSeries, the state of a queue manager prior to it being stopped. In this state, programs are allowed to finish processing, but no new programs are allowed to start.

R

reason code. A return code that describes the reason for the failure or partial success of an MQI call.

receiver channel. In message queuing, a channel that responds to a sender channel, takes messages from a communication link, and puts them on a local queue.

relative byte address (RBA). The displacement in bytes of a stored record or control interval from the beginning of the storage space allocated to the data set to which it belongs.

remote queue. A queue belonging to a remote queue manager. Programs can put messages on remote queues, but they cannot get messages from remote queues. Contrast with *local queue*.

remote queue manager. To a program, a queue manager that is not the one to which the program is connected.

remote queue object. See *local definition of a remote queue*.

remote queuing. In message queuing, the provision of services to enable applications to put messages on queues belonging to other queue managers.

reply message. A type of message used for replies to request messages.

reply-to queue. The name of a queue to which the program that issued an MQPUT call wants a reply message or report message sent.

report message. A type of message that gives information about another message. A report message can indicate that a message has been delivered, has arrived at its destination, has expired, or could not be processed for some reason.

requester channel. In message queuing, a channel that may be started remotely by a sender channel. The

request message • system initialization table (SIT)

requester channel accepts messages from the sender channel over a communication link and puts the messages on the local queue designated in the message. See also *server channel*.

request message. A type of message used to request a reply from another program.

resolution path. The set of queues that are opened when an application specifies an alias or a remote queue on input to an MQOPEN call.

resource. Any facility of the computing system or operating system required by a job or task.

resource manager. An application, program, or transaction that manages and controls access to shared resources such as memory buffers and data sets. MQSeries, CICS, and IMS are resource managers.

responder. In distributed queuing, a program that replies to network connection requests from another system.

resynch. In MQSeries, an option to direct a channel to start up and resolve any in-doubt status messages, but without restarting message transfer.

return codes. The collective name for completion codes and reason codes.

rollback. Synonym for *back out*.

RTM. Recovery termination manager.

S

sender channel. In message queuing, a channel that initiates transfers, removes messages from a transmission queue, and moves them over a communication link to a receiver or requester channel.

sequential delivery. In MQSeries, a method of transmitting messages with a sequence number so that the receiving channel can reestablish the message sequence when storing the messages. This is required where messages must be delivered only once, and in the correct order.

sequential number wrap value. In MQSeries, a method of ensuring that both ends of a communication link reset their current message sequence numbers at the same time. Transmitting messages with a sequence number ensures that the receiving channel can reestablish the message sequence when storing the messages.

server. (1) In MQSeries, a queue manager that provides queue services to client applications running

on a remote workstation. (2) The program that responds to requests for information in the particular two-program, information-flow model of client/server. See also *client*.

server channel. In message queuing, a channel that responds to a requester channel, removes messages from a transmission queue, and moves them over a communication link to the requester channel.

server connection channel type. The type of MQI channel definition associated with the server that runs a queue manager. See also *client connection channel type*.

session ID. In MQSeries for MVS/ESA, the CICS-unique identifier that defines the communication link to be used by a message channel agent when moving messages from a transmission queue to a link.

shutdown. See *immediate shutdown*, *preemptive shutdown*, and *quiesced shutdown*.

single-phase backout. A method in which an action in progress must not be allowed to finish, and all changes that are part of that action must be undone.

single-phase commit. A method in which a program can commit updates to a queue without coordinating those updates with updates the program has made to resources controlled by another resource manager. Contrast with *two-phase commit*.

Socket. A communications handle used by TCP/IP programs to send and receive data to and from a remote host.

subsystem. In MVS, a group of modules that provides function that is dependent on MVS. For example, MQSeries for MVS/ESA is an MVS subsystem.

symptom string. Diagnostic information displayed in a structured format designed for searching the IBM software support database.

synchronous messaging. A method of communication between programs in which programs place messages on message queues. With synchronous messaging, the sending program waits for a reply to its message before resuming its own processing. Contrast with *asynchronous messaging*.

syncpoint. An intermediate or end point during processing of a transaction at which the transaction's protected resources are consistent. At a syncpoint, changes to the resources can safely be committed, or they can be backed out to the previous syncpoint.

system initialization table (SIT). A table containing parameters used by CICS on start up.

T

target library high-level qualifier (thlqual). High-level qualifier for MVS/ESA target data set names.

task switching. The overlapping of I/O operations and processing between several tasks.

TCP/IP.. Transmission Control Protocol, Internet Protocol. TCP/IP is a family of communications protocols.

termination notification. A pending event that is activated when a CICS subsystem successfully connects to MQSeries for MVS/ESA.

thlqual. Target library high-level qualifier.

thread. In MQSeries, the lowest level of parallel execution available on an operating system platform.

time-independent messaging. See *asynchronous messaging*.

trace. In MQSeries, a facility for recording MQSeries activity. The destinations for trace entries can include GTF and the system management facility (SMF).

tranid. See *transaction identifier*.

transaction identifier. In CICS, a name that is specified when the transaction is defined, and that is used to invoke the transaction.

transmission program. See *message channel agent*.

transmission queue. A local queue on which prepared messages destined for a remote queue manager are temporarily stored.

triggering. In MQSeries, a facility allowing a queue manager to start an application automatically when predetermined conditions on a queue are satisfied.

trigger message. A message containing information about the program that a trigger monitor is to start.

trigger monitor. A continuously-running application serving one or more initiation queues. When a trigger message arrives on an initiation queue, the trigger monitor retrieves the message. It uses the information in the trigger message to start a process that serves the queue on which a trigger event occurred.

two-phase commit. A protocol for the coordination of changes to recoverable resources when more than one resource manager is used by a single transaction. Contrast with *single-phase commit*.

U

undelivered-message queue. See *dead-letter queue*.

unit of recovery. A recoverable sequence of operations within a single resource manager. Contrast with *unit of work*.

unit of work. A recoverable sequence of operations performed by an application between two points of consistency. A unit of work begins when a transaction starts or after a user-requested syncpoint. It ends either at a user-requested syncpoint or at the end of a transaction. Contrast with *unit of recovery*.

utility. In MQSeries, a supplied set of programs that provide the system operator or system administrator with facilities in addition to those provided by the MQSeries commands. Some utilities invoke more than one function.

Index

A

- alias queue manager, creating 57
- alias queues
 - creating 56
 - description 5
- alias reply, creating 58
- allocating MQSeries files when installing 12
- application
 - data 1
 - design considerations 105
 - MQI administration support 137
 - programming errors, examples of 98
 - time-independent 1
- applications
 - C programming language 138
 - design guidelines 138
 - PL/I programming language 138
- attributes of queues 4
- automatic reorganization 53

B

- batch interface
 - data integrity 89
 - how to use 88
 - restrictions on use 90
 - sample programs 88
 - using 88
 - verifying 89
- batch modules 86
- bibliography x
- BookManager xiv
- browse function 80
- building applications 137

C

- channel
 - configuration 190
 - definitions
 - creating 60
 - modifying 63
 - description 6
 - monitoring 78
 - opening and closing 69
- CICS
 - connection definition 31
 - file management 14
 - initialization PLT (PLTPI) list 24
 - installing table entries for MQSeries 13
 - journal control table 15
 - modifying start-up deck 14

- CICS (*continued*)
 - Program List Table Shut Down (PLTSD) 24
 - recovery and restart 14
 - session definition 32
 - start up and shutdown 24
- client
 - configuration support 6
 - error messages on DOS and Windows 110
 - problem determination 109
 - support for 199
- close object function call 126
- code-page conversion
 - genxlt utility 200
 - iconv functions 200
 - iconv utility 200
- command line function 83
- communications process 81
- configuration functions
 - channel definitions
 - creating 60
 - modifying 63
 - display options
 - channel definitions 65
 - global system definition 65
 - queue definitions 65
 - global system definition 45
 - main menu 44
 - queue definitions, creating
 - alias queue 56
 - alias queue manager 57
 - alias reply 58
 - local queue 49
 - local queue extended definition 50
 - main screen 48
 - remote queue 55
 - queue definitions, modifying and deleting
 - deleting an existing queue 60
 - modifying an existing queue 60
 - selecting an existing queue 59
- configuration guidelines
 - channel 35
 - example of 39
 - overview 33
 - queue 38
 - queue manager 33
- configuration worksheet 185
- configuring network communications 29
- connect queue manager function call 126
- Correlld, performance considerations when using 105
- creating
 - alias queue manager 57
 - alias queues 56

Index

creating (*continued*)
 alias reply 58
 channel definitions 60
 local queue extended definition 50
 local queues 49
 queue manager 44
 remote queues 55

D

data types, detailed description
 MQDLH structure 119
 MQGMO structure 120
 MQMD structure 120
 MQOD structure 123
 MQPMO structure 124
 MQTM structure 124
dead-letter header structure 119
dead-letter queue
 description 5
 specifying 45
debugging
 common programming errors 98
 preliminary checks 95
defining installation configuration resources 18
defining queues 4
delete all function 91
deleting
 channel definitions 63
 queue definitions 60
disconnect queue manager function call 127
displaying
 channel definitions 65
 global system definition 65
 queue definitions 65
distributed queuing
 dead-letter queue 5
 incorrect output 108
 undelivered-message queue 5
DOS clients error messages 110

E

error logs 81
event queue 6
event-driven processing 1
examples, programming errors 98

F

function calls, detailed description
 MQCLOSE 126
 MQCONN 126
 MQDISC 127
 MQGET 128
 MQINQ 129

function calls, detailed description (*continued*)
 MQOPEN 131
 MQPUT 132
 MQPUT1 133
 MQSET 134
function key operations 43

G

genxlt utility 200
get message options structure 120
get queue manager function call 128
global system definition
 entry fields 45
 guidelines
 backing up configuration file 48
 queue manager name 45
 specifying a dead-letter queue 45
glossary 221
guidelines, product configuration 33

H

HTML (Hypertext Markup Language) xiv
Hypertext Markup Language (HTML) xiv

I

iconv utility 200
incorrect output 106
Information Presentation Facility (IPF) xiv
initializing MQSeries for VSE/ESA 16
initializing system 72
inquire object attributes function call 129
installation
 allocating MQSeries files 12
 CICS start up and shutdown 24
 defining local queues 17
 defining SYSTEM.LOG queue 18
 initializing the system 16
 local queue verification test 19
 migration procedure, steps required 24
 modifying CICS start-up deck 14
 preparing CICS table entries for MQSeries 13
 procedures for new users 12
 product 10
 restoring MQSeries library 11
 samples 9
 specify the queue manager name 16
 starting MQSeries for VSE/ESA 15
 tape contents 9
 target installation library 10
 uppercase translation 15
 verification test 19
 VSAM installation catalog 10

intercommunication example 185—198
 IPF (Information Presentation Facility) xiv

L

library tape 9
 local queue extended definition, creating 50
 local queues
 creating 49
 dead-letter 5
 defining during installation 17
 description 5
 transmission 5
 undelivered-message 5
 log, system 108
 LU 6.2 connection 185

M

main configuration menu 44
 maintenance, queue 73
 master terminal 42
 message
 containing unexpected information 107
 description 1
 descriptor 1
 errors on DOS and Windows clients 110
 information 202
 lengths of 2
 not appearing on queues 106
 performance affected by length 105
 queuing 1
 retrieval algorithms 2
 searching for particular 105
 undelivered 109
 variable length 105
 message channel agent 81
 message descriptor structure 120
 message queue interface (MQI) 1
 message queuing 1
 message sequence number, resetting 70
 message-driven processing 1
 migrating from an earlier release 24
 modifying
 channel definitions 63
 queue definitions 60
 monitoring functions
 channels 78
 main menu 75
 queues 75
 MQCL 83
 MQCLOSE call 126
 MQCONN call 126
 MQDATA 110
 MQDISC call 127

MQDLH structure 119
 MQGET call 128
 MQGMO structure 120
 MQI
 description 1
 local administration support 137
 queue manager calls 4
 MQINQ call 129
 MQIT transaction 16
 MQJCONFIG.Z 12
 MQJQUEUE.Z 12
 MQJSETUP.Z 12
 MQMD structure 120
 MQMT 42
 MQOD structure 123
 MQOPEN call 131
 MQPMO structure 124
 MQPREORG function 91
 MQPUT and MQPUT1, performance considerations 106
 MQPUT call 132
 MQPUT1 call 133
 MQPUTIL 86
 MQSE transaction 16
 MQSeries for VSE/ESA
 allocating files when installing 12
 building applications 137
 channel configuration 190
 channel definition 35
 client support 199
 command line function 83
 configuration 190
 configuration worksheet 185
 definition in CICS 30
 LU 6.2 connection 185
 overview 111
 running applications 137
 MQSeries publications x
 MQSET call 134
 MQTM structure 124
 Msgld, performance considerations when using 105

N

names of objects 3
 network configuration 29
 new users, procedures for 12

O

object descriptor structure 123
 objects
 names 3
 queue 5
 queue manager in MQI calls 4
 types 3

Index

- open object function call 131
- opening and closing channels 69
- operations functions
 - open and close channels 69
 - queue maintenance 73
 - reset message sequence number 70
 - start and stop queues 66
 - system initialization 72
- operator function keys 43
- overview of MQSeries for VSE/ESA 111

P

- panel layout 42
- performance considerations when using trace 109
- permanent queues 2
- PLTPI 24
- PLTSD 24
- PostScript format xiv
- predefined queues 2
- problem determination
 - clients 109
 - incorrect output
 - messages containing unexpected information 107
 - messages not appearing on queues with distributed queuing 106
 - network operation
 - SNA problems 96
 - TCP/IP problems 97
 - programming errors 98
 - trace 109
 - with local queue operation 95
- procedures (installation) for new users 12
- processing, event-driven 1
- product configuration 33
- programming errors, examples of 98
- publications, MQSeries x
- publications, related xv
- put message function call 132
- put message options structure 124
- put one message function call 133

Q

- queue
 - alias 5
 - attributes of 4
 - dead-letter, specifying 45
 - defining 4
 - description 2
 - distributed, incorrect output from 108
 - event 6
 - for MQSeries applications 137
 - monitoring 75
 - objects
 - alias 5

- queue (*continued*)
 - objects (*continued*)
 - local 5
 - remote 5
 - predefined 2
 - remote 5
 - starting and stopping 66
 - transmission 5
- queue definition
 - deleting 60
 - main screen 48
 - modifying 60
 - selecting 59
- queue maintenance 73
- queue manager
 - configuration file, backing up 48
 - creating 44
 - description 4
 - name, specifying during installation 16
 - object in MQI calls 4
 - unique name 45

R

- recovery 14
- related publications xv
- remote queues
 - creating 55
 - description 5
- reorganization, automatic 53
- reorganizing queue files 92
- requirements
 - hardware 111
 - software 111
- resetting message sequence number 70
- restoring installation library 11
- retrieval algorithms for messages 2
- return codes 98
- running applications 137

S

- sample code 138
- sample JCL 93
- selecting queue definitions 59
- selecting transmission protocol 61
- set object attributes function call 134
- softcopy books xiv
- specified operating environment 111
- starting and stopping queues 66
- starting CICS 24
- starting MQSeries 15
- syncpoints and triggers 139
- system initialization 72
- system log 108

system operation 41
SYSTEM.LOG queue, defining 18

T

tape contents 9
target installation library 10
TCP/IP
 defining a receiver channel 198
 defining a sender channel 197
 establishing a connection 190
 installation requirements 29
 listener port 47
 selecting 61
terminology used in this book 221
time-independent applications 1
trace, performance considerations 109
transactional interface 83
transmission protocol, selecting 61
transmission queue 5
trigger message structure 124
triggering 141
types of objects 3

U

undelivered message queue
 See dead-letter queue
unit of work 139
uppercase translation 15
using the batch interface 88

V

verification, installation 19
VSAM file maintenance
 creating space 90
 delete all function 91
 MQPREORG function
 description of 91
 sample JCL 93
 reorganizing queue files 92
VSAM installation catalog 10
VSE/ESA system requirements 111

W

Windows clients error messages 110
Windows Help xiv

Sending your comments to IBM

MQSeries® for VSE/ESA

System Management Guide

GC34-5364-00

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 request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should 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, use the Readers' Comment Form
- By fax:
 - From outside the U.K., after your international access code use 44 1962 870229
 - From within the U.K., use 01962 870229
- Electronically, use the appropriate network ID:
 - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
 - IBMLink: HURSLEY(IDRCF)
 - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

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

Readers' Comments

MQSeries® for VSE/ESA

System Management Guide

GC34-5364-00

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer. This form is provided for comments about the information in this manual and the way it is presented.

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.

Be sure to print your name and address below if you would like a reply.

Name

Address

Company or Organization

Telephone

Email



You can send your comments POST FREE on this form from any one of these countries:

Australia	Finland	Iceland	Netherlands	Singapore	United States
Belgium	France	Israel	New Zealand	Spain	of America
Bermuda	Germany	Italy	Norway	Sweden	
Cyprus	Greece	Luxembourg	Portugal	Switzerland	
Denmark	Hong Kong	Monaco	Republic of Ireland	United Arab Emirates	

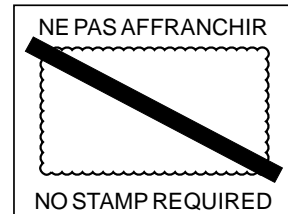
If your country is not listed here, your local IBM representative will be pleased to forward your comments to us. Or you can pay the postage and send the form direct to IBM (this includes mailing in the U.K.).

1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCR NUMBER: PHQ-D/1348/SO



REPONSE PAYEE
GRANDE-BRETAGNE

IBM United Kingdom Laboratories
Information Development Department (MP095)
Hursley Park,
WINCHESTER, Hants
SO21 2ZZ United Kingdom

3 Fold along this line

From: Name _____
Company or Organization _____
Address _____

EMAIL _____
Telephone _____

1 Cut along this line

4 Fasten here with adhesive tape



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

GC34-5364-00





MQSeries® for VSE/ESA

System Management Guide

Version 2 Release 1