MQSeries for VSE/ESA

SC33-1142-02

User's Guide

Version 1 Release 4

Note!
Before using this information and the product it supports, be sure to read the general information under “Notices” on page xv.

Third Edition (March 1997)

This edition applies to Version 1 Release 4 of IBM MQSeries for VSE/ESA (program number 5787-ECX) and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This book is based on Version 1 Release 3.1, order number SC33-1142-01. Changes from that edition are marked by vertical lines to the left of the text.

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, SO21 2JN, United Kingdom

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.

Contents

Figures

Tables

Notices

The following paragraph does not apply to any 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 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.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. 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 of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

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:

Laboratory Counsel
Mail Point 151
IBM United Kingdom Laboratories
Hursley Park, Winchester
Hampshire, SO21 2JN
U.K.

Such information may be available, subject to appropriate terms and conditions, including, in some cases, payment of a fee.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing to The IBM Director of Licensing, IBM Corporation, 500 Columbus Ave, Thornwood, New York, 10594, U.S.A.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX	CICS	IBM
MQ	MQSeries	MVS/ESA
OS/2	OS/400	RACF
RISC System/6000	VSE	VTAM

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

C-bus is a trademark of Corollary, Inc.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

About this book

The purpose of this User's Guide is to provide all information necessary for a user to install IBM MQSeries for VSE/ESA software, and to fully utilize its features to provide the communications framework for distributed applications based on the IBM Message Queue Interface (MQI).

To accomplish this goal, this guide describes the IBM MQSeries for VSE/ESA software; its installation, configuration, and operations; and the programming interface to be used by the developers of applications.

Throughout this document, IBM MQSeries for VSE/ESA is referred to simply as MQSeries System.

Who should use this book

The introductory product description sections of this guide will be of interest to *all users*. Beyond that, different portions of this guide are intended for different audiences.

System or *Network Administrators* responsible for installing, operating and maintaining MQSeries System software will be primarily interested in Chapters 1 through 6.

Distributed Application Designers will be interested in Chapters 3 through 7.

Application Developers will be primarily interested in Chapter 7.

What's in this book

This guide provides information about the MQSeries System software as implemented for VSE/ESA systems.

How to use this book

This User's Guide consists of seven chapters and six appendixes organized as follows:

- Chapter 1, "Product description" on page 1—describes the MQSeries System and services, provides an overview of the components and architecture, and provides an application example.
- Chapter 2, "Installation" on page 3—highlights the system requirements for using the MQSeries System software and provides a detailed procedure for installing the software.
- Chapter 3, "Planning" on page 15—provides an overview of the considerations for implementing a distributed application using the MQSeries System.
- Chapter 4, "Configuration" on page 23—covers the details for creating the system services to support your application.
- Chapter 5, "Configuring network resources" on page 51—provides guidance and general help in configuring the network resources to enable the MQSeries System to function.
- Chapter 6, "System operation" on page 65—provides procedures for activating system services and troubleshooting system problems.
- Chapter 7, "Application programming interface" on page 105—provides an alphabetical reference of the application programming interface calls.
- Appendix A, "System messages" on page 155—lists internal messages generated when application programs activating the MQSeries System encounter abnormal conditions, and external messages generated by the Administration programs.
- Appendix B, "COBOL programming language examples" on page 189—contains a description of the supplied COBOL copy files, program structures, and program call examples.

- Appendix C, “CICS control table definitions” on page 197—contains a listing of the current CICS Control Table Definition entries for the CICS/MQSeries System Subsystem.
- Appendix D, “Sample programs” on page 207—contains a sample program ECHO that can be used to test MQSeries System configurations.
- Appendix E, “COBOL copybooks” on page 269—contains a listing of the supplied COBOL Copybook files.
- Appendix F, “Configuration worksheets” on page 277—contains blank worksheets to aid in the design and planning of a distributed application using the MQSeries System.

Note: Only the “Value” copybooks are in Appendix E, “COBOL copybooks” on page 269. Additional “Linkage” copybooks will be in the installation user library with an “L” suffix instead of a “V” suffix.

Typographical conventions

boldface
Identifies an item in an MQSeries System window. The item could be a keyword, an action, a field label, or a pushbutton. Whenever one of the steps in a procedure includes a word in boldface, look for an item in the window that is labeled with that word.

bold italics
Are used for emphasis. ***Take extra care*** wherever you see bold italics!

italics
Identify one of the following:

- New terms that describe MQSeries System components or concepts. A term printed in italics is usually followed by its definition.
- Parameters for which you supply the actual names or values.
- References to other books.

<angle brackets>
Identify a key on the keyboard. The instruction “press <Enter>” means “Find the key labeled ‘Enter’ and press it.” If the instruction identifies two (or more) keys, hold down the first key while you press the second key.

`monospace`
Identifies one of the following:

- Text as shown, make sure you type the uppercase and lowercase characters exactly as shown.
- Names of files and directories (path names).

Where to find more information

MQSeries publications

Evaluating products

IBM MQSeries Brochure, G511-1908

IBM MQSeries: An Introduction to Messaging and Queuing, GC33-0805

IBM MQSeries: Concepts and Architecture, GC33-1141

IBM MQSeries Message Queue Interface Technical Reference, SC33-0850

Planning

IBM MQSeries Planning Guide, GC33-1349

IBM MQSeries for MVS/ESA Version 1 Release 1.4 Licensed Program Specifications, GC33-1350

IBM MQSeries for OS/400 Version 3 Release 2 (and later) Licensed Program Specifications, GC33-1360 (softcopy only)

Administration

IBM MQSeries Programmable System Management, SC33-1482

IBM MQSeries Command Reference, SC33-1369

IBM MQSeries for AIX Version 2 Release 2.1 System Management Guide, SC33-1373

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

IBM MQSeries for HP-UX Version 2 Release 2.1 System Management Guide, GC33-1633

IBM MQSeries for MVS/ESA Version 1 Release 1.4 Program Directory, GC33-1626

IBM MQSeries for MVS/ESA Version 1 Release 1.4 System Management Guide, SC33-0806

IBM MQSeries for OS/2 Version 2.0.1 System Management Guide, SC33-1371

IBM MQSeries for OS/400 Version 3 Release 2 (and later) Administration Guide, GC33-1361

IBM MQSeries for SunOS Version 2.2 System Management Guide, GC33-1772

IBM MQSeries for Sun Solaris Version 2.2 System Management Guide, GC33-1800

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

IBM MQSeries for Windows NT Version 2 Release 0 System Management Guide, SC33-1643

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

IBM MQSeries link for R/3 Version 1.0 User's Guide, GC33-1934

Application programming

IBM MQSeries Application Programming Guide, SC33-0807

IBM MQSeries Application Programming Reference, SC33-1673

IBM MQSeries Application Programming Summary, SX33-6095

IBM MQSeries for OS/400 Version 3 Release 2 (and later) Application Programming Reference (RPG), SC33-1362

Problem determination

IBM MQSeries for MVS/ESA Version 1 Release 1.4 Problem Determination Guide, SC33-0808

IBM MQSeries for MVS/ESA Version 1 Release 1.4 Messages and Codes, SC33-0819

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

Special topics

IBM MQSeries Distributed Queuing Guide, SC33-1139

IBM MQSeries Clients, GC33-1632

Other MQSeries publications

For information about other MQSeries platforms, see the following publications:

IBM MQSeries for AT&T GIS UNIX User's Guide, SC33-1437

IBM MQSeries for Digital VMS VAX User's Guide, SC33-1144

IBM MQSeries for HP-UX User's Guide, SC33-1376

IBM MQSeries for OS/400 User's Guide, SC33-1145

IBM MQSeries for SCO UNIX User's Guide, SC33-1378

IBM MQSeries for SunOS User's Guide, SC33-1377

IBM MQSeries for Sun Solaris User's Guide, SC33-1439

IBM MQSeries for Tandem NonStop Kernel, SC33-1755

IBM MQSeries for UnixWare User's Guide, SC33-1379

IBM MQSeries for VSE/ESA User's Guide, SC33-1142

What's new with MQSeries for VSE 1.4

- COBOL for VSE with LE support
- Full 31 bit addressing
- Reorganized batch utility
- Case sensitive queue naming
- Inbound ping request support
- Y2000 compliance
- Standard MSHP library install support
- 64 character non-CICS TP names
- Additional code page support for communication with other platforms
- Dual queue support

Chapter 1. Product description

IBM MQSeries for VSE/ESA enables application programs to exchange messages with other CICS applications and with remote MQSeries applications running on systems such as other IBM Mainframes, VAXs**, Tandems**, PC LANs, etc.

The MQSeries System provides a set of messaging and queuing services which support data transfer between distributed applications. These services allow applications to communicate without knowledge of the lower levels of the communications network and without specific knowledge of the location of the other applications. The messaging and queuing services are accessed via the IBM Message Queuing Interface (MQI).

Version 1 MQSeries System elements

There are four key conceptual elements within the MQSeries System which must be well understood. They are *messages*, *queues*, *queue managers*, and *channels*.

Messages

All data transferred by the MQSeries System is in the form of a *message* exchanged between cooperating distributed applications. Every message has two parts. The body of the message contains the *user data* supplied by an application. This user data is never touched by the MQSeries System.

Ancillary data commonly called a *header,* is added to the message by the MQSeries System to provide routing and other control information required for message delivery. The header is not normally seen by the application programs.

Messages are exchanged between applications via *queues*.

Queues

A *message queue* is simply a disk file used by the MQSeries System to hold messages. The physical management of queues is entirely hidden from the application programs. Applications have no access to the queues other than through the message queuing interface, MQI.

Message queues are classified as either *local* or *remote.* These terms are defined from an application perspective. A *local queue* is any queue residing on the same message queuing system as the application. A *remote queue* is any queue residing on another message queuing system.

The special case of a local queue which is used to hold messages to be transmitted to another system is called a *transmission queue*.

An *alias queue* is not a true physical queue, but rather a logical naming capability which allows an alias queue name to be resolved to another real queue, either local or remote. This provides a mechanism for logical indirection which often proves a convenient method to allow application programs to be completely independent of the underlying message queuing definitions.

The physical management of the queues is provided by the *queue manager*.

Queue manager

The queue manager is responsible for providing the message queuing services used by applications. Applications access these services by using the MQI calls to communicate with the local queue manager (the queue manager on the same system as the application). It is most common to think of a queue manager as having a one-to-one correspondence to an MQSeries System installation. That is, normally there is one queue manager per system.

Channels

A *channel* is a unidirectional point-to-point communications link between two MQSeries Systems. Messages flow over a channel in one direction only. If two MQSeries Systems need to *exchange* messages, then two channels are required.

For outbound channels, the MQSeries System reads messages from the associated *transmission queue* and sends them to the remote system via the communications channel. For inbound channels, the MQSeries System receives messages from the communication link and writes them to the destination *local* queue.

Software components of the MQSeries System

The MQSeries System consists of the following software components:

Message queue interface (MQI)

The Version 1 MQSeries System implementation of MQI is built around static COBOL calls to MQI verbs (see Chapter 7, “Application programming interface” on page 105). It is responsible for handling user application requests to read and write from the queuing system, and for arbitrating among multiple requests to the same queue. The MQI functions are provided in the form of members of the VSE/ESA object library. Appropriate MQI functions are link edited into application programs that use MQSeries System services.

Message channel agent (MCA)

The Message Channel Agent (MCA) consists of a set of CICS transactions which implement the Message Channel Protocol (MCP). The MCP is the high level protocol used to transport messages between MQSeries Systems. This protocol is implemented on top of an industry standard transport layer protocol (TLP) LU6.2. The underlying TLP is not provided with the MQSeries System but is a prerequisite.

Message queue management (MQM)

The Version 1 MQSeries System Administration and Operations functions are menu driven. They allow the system administrator to define, modify, and delete MQSeries System queues, aliases, and channels; and to perform various maintenance tasks such as resetting message sequence numbers, purging queues, and monitoring the status of the MQSeries System.

System monitor

This long running task controls recovery, triggering, and quiescing. Refer to “In-storage-control-blocks and recovery mechanism” on page 284 for more details.

Sample programs

Source code and phase modules for four sample application programs is provided. These are test programs which will be used in verifying the system installation and which may also be referred to for examples of MQI calls.

These programs include TTPTST1, TTPTST2, TTPTST3, and MQPECHO which are supplied on the installation tape. Listings of these four programs may be found in Appendix D, “Sample programs” on page 207.

Chapter 2. Installation

This chapter highlights the system requirements for using the MQSeries System software and provides a detailed procedure for installing the software.

Prerequisites for normal operation

The MQSeries System has specific software requirements that must be met for proper operation. They are:

Hardware

- Any IBM System 370 or 390
  - 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
- Any communications hardware supporting SNA/LU6.2

Software

- VSE/ESA 1.4 (5750-ACD) or later 1.x
- CICS/VSE 2.3 (5686-026) or later 2.x
- VTAM for VSE/ESA 4.2 (5666-363) or later 4.x
- LE/VSE 1.4 Runtime library

Supported language for application development

- COBOL for VSE

Migration guidance

- Redefinition of channels is not required
- Redefinition of queues is not required
- Redefinition of Configuration file is needed
- The software levels of VSE/ESA and CICS/VSE listed above are prerequisites
- Recompilation of customer applications in LE/VSE is required

Contents of the distribution tape

The distribution tape consists of only one VSE/ESA sublibrary in MSHP format. The original sublibrary name was “PRD2.MQSERIES”, and we strongly suggest you use the same sublibrary name when you restore it. This sublibrary contains phases, object decks, copybooks and samples.

- Copy books are to be used by your CICS applications whenever you intend to call the MQSeries Application Programming Interface (API).
- Object decks will be called at linkedit time when you are building your own MQSeries applications (autolink).
- Phases were all compiled in COBOL for VSE with LE/VSE and linkedited with AMODE(31) and RMODE(ANY).

– Samples have Z as member type. Some of them need to be modified for the VSE/POWER JECL statements:

```
* ** JOB      to      * $$ JOB
* ** LST      to      * $$ LST
* ** SLI      to      * $$ SLI
* ** EOJ      to      * $$ EOJ
```

Here are short descriptions of these samples:

MQJCONFG.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
MQCICSDT.Z	Entry definitions for CICS DCT.
MQCICSFT.Z	Entry definitions for CICS FCT.

Note: Migration jobs are to migrate an MQSeries for VSE/ESA 1.3 to this new 1.4 release.

MQSeries System installation

The steps to accomplish the MQSeries System Installation are contained in the following procedure:

Installing The MQSeries system

1. Create a VSAM user catalog. This is an optional step. It is recommended that the user use the Interactive Interface Dialogs (II) to create this catalog. In the examples VSAM catalog named MQMCAT is being used, and it is assumed that its label is already defined in the Disk Label Area.
2. Allocate a VSE library.
   This step is not required if you restore the product into the PRD2 library which is standard in a VSE/ESA system. However, for various reasons, you may want to install MQSeries in another library. It is recommended you use the Interactive Interface dialogs for creating this library, or run the sample below adapted for your environment. Also, don't forget to modify provided samples accordingly.

```
* $$ JOB JNM=DEFLIB,CLASS=O,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)
/*
```

```
/&
* $$ EOJ
```

3. Restore the MQSeries sublibrary from the distribution tape. There are 2 ways of doing the restore:

   1. by using the Interactive Interface Dialogs:
      - from the administrator panel select: Installation
      - select STacked V2 Format
      - select “prepare for installation”
      - submit the job, and wait for its completion.
      - select “Install Product(s) from Tape”; entering PRD2.MQSERIES for the sublibrary name, or the one you have defined if you don't use the default one.
   2. by using the following job:

```
* $$ JOB JNM=MQMTAPE,CLASS=0,DISP=D
// JOB MQMTAPE Restore MQSeries from tape
// ASSGN SYS006,cuu
// MTC REW,SYS006
// EXEC MSHP,SIZE=1M
INSTALL PRODUCT FROM TAPE ID='MQSeries 1.4.0'-
PROD INTO=lib.sublib
/*
/&
* $$ EOJ
```

   Where:

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

Allocation and initialization of subsystem files for new users

Warning: For users upgrading from IBM MQSeries for VSE/ESA Version 1 Release 3, please use the procedure “Re-initialization of subsystem files from Version 1.3” on page 6.

Customize and submit the following jobs contained in PRD2.MQSERIES to allocate and initialize the MQSeries System (CICS) subsystem files:

- **MQJSETUP.Z**- Allocates the MQSeries System (CICS) subsystem setup file. It is used to populate the configuration file of textual information used by the runtime MQSeries System.
- **MQJCONFG.Z** - Allocates the MQSeries System (CICS) subsystem configuration file. For this VSAM KSDS file, each record is a fixed length of approximately 2K bytes. To estimate the space required, one record is needed for the following MQSeries System (CICS) subsystem objects:
  - Each channel
  - Each object

  A space allocation of one cylinder is sufficient for normal installations.
- **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 System (CICS) subsystem.

  To estimate the space required for each message queue, use the following guidelines:
  - Each message queue file contains one header record per local queue.
  - One record will be written per user message.
  - Each record is variable length and consists of a header, plus the actual variable length user data area. Each record header is 736 bytes.
  - 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

Note: The above three files need to be INITIALIZED.

The following are sample definitions for user message queue files:

MQSERIES.MQFI001

MQSERIES.MQFO001

MQSERIES.MQFI002

MQSERIES.MQFO002

MQSERIES.MQFI003

MQSERIES.MQFO003

Note: Multiple local queues can be defined in one physical file. However, this is recommended only for low activity queues.

Re-initialization of subsystem files from Version 1.3

- Invariants from 1.3:
  If you are upgrading from version 1.3 of MQSeries for VSE, you may keep every file except MQFSSET and MQFCNFG. You may also keep the same CICS start-up deck and entries in the two tables: FCT and DCT.
- Customize MQFSSET as described in “Allocation and initialization of subsystem files for new users” on page 5 by executing MQJSETUP.Z.
- Backup your old configuration file, MQFCNFG.
- Save your system definition, channel definitions and queue definitions by executing MQJMIGR1.Z against MQFCNFG to create MQSERIES.MQOCNFG.
- Run MQJCONFG.Z to create a new MQFCNFG.
- Bring up CICS and execute the MQSU transaction.
- Close MQFCNFG using CEMT.
- Reproduce MQSERIES.MQOCNFG into MQFCNFG by executing MQJMIGR2.Z.
- Open MQFCNFG using CEMT.
- Use the 1.1 option of MQMT to override the default values of Max Recovery Tasks and System Wait Interval if needed.
- Verify correctness of all queue and channel definitions before deleting the old configuration.

MQJMIGR1 sample JCL

```
* ** JOB JNM=MQJMIGR1,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJMIGR1 - Migrate MQSeries 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 filed with ?volid? have also to be modified to suit the  *
*  user specifications.                                             *
*                                                                   *
* ------------------------------------------------------------------*
*                                                                   *
```

```
*   Use this sample only to migrate from version 1.3 to 1.4.        *
*                                                                    *
*   This job  extracts your system, queue and channel definitions    *
*   from a version 1.3 configuration file.                           *
*   It then reformats them into the new version 1.4 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                               *
*                                                                    *
* 5787-ECX                                                           *
* (C) Copyright IBM Corp. 1993, 1996                                 *
*                                                                    *
* US Government Users Restricted Rights - Use, duplication or        *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp.  *
* -------------------------------------------------------------------*
*
// DLBL OLDCNFG,'MQSERIES.MQFCNFG',,VSAM,CAT=MQMCAT
// DLBL NEWCNFG,'MQSERIES.MQOCNFG',,VSAM,CAT=MQMCAT
// 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?)
          SET MAXCC = 0
          DEFINE CLUSTER                                       -
                   (NAME (MQSERIES.MQOCNFG)                    -
                    RECORDS (50 10)                            -
                    RECORDSIZE (2048 2048)                     -
                    VOLUMES (?volid?)                          -
                    KEYS    (100 0)                            -
                    SHR     (2)                                -
                    INDEXED)                                   -
                  DATA                                         -
                    (NAME (MQSERIES.MQOCNFG.DATA) CISZ(4096)) -
                  INDEX                                        -
                    (NAME (MQSERIES.MQOCNFG.INDEX) CISZ(512)) -
                     CAT (?CAT?)
/*
// IF $MRC > 0 THEN
// GOTO NOPROC
// LIBDEF PHASE,SEARCH=(PRD2.MQSERIES,PRD2.SCEEBASE)
// EXEC MQPCONFG,SIZE=AUTO
/*
/. NOPROC
/&
* ** EOJ
```

MQJMIGR2 sample JCL

```
* ** JOB JNM=MQJMIGR2,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJMIGR2 - Migrate MQSeries 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"                        *
* ------------------------------------------------------------------*
*                                                                   *
*   Use this sample only to migrate from version 1.3 to 1.4.        *
*                                                                   *
*  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                              *
*                                                                   *
* 5787-ECX                                                          *
* (C) Copyright IBM Corp. 1993, 1996                                *
*                                                                   *
* US Government Users Restricted Rights - Use, duplication or       *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
* ------------------------------------------------------------------*
// DLBL MQFCNFG,'MQSERIES.MQFCNFG',,VSAM,CAT=MQMCAT
// DLBL MQOCNFG,'MQSERIES.MQOCNFG',,VSAM,CAT=MQMCAT
// 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?)
/*
/&
* ** EOJ
```

Install CICS table entries

Use the samples (see Appendix D, “Sample programs”, on page 207) provided with the product. Refer to Appendix C, “CICS control table definitions”, on page 197 for additional information.

To help you install the PCT and PPT CICS definitions, the sample MQJCSD.Z is provided. It 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, since all entries are defined in group “MQM” which is then added to the VSELIST list.

Note if migrating from MQSeries 1.3:

If you are migrating, from release 1.3, we suggest you suppress all entries from the CSD (if defined) before running this job or, if you use PPT and PCT tables, to remove the MQSeries entries and recompile these tables.

- File Control Table (FCT) - sample entries are defined in member MQCICSFT.Z. Please review this member for further details.
- Destination Control Table (DCT) - The product requires intra partition transient data queues CSMT and MQER. See the sample DCT defined in member MQCICSDT.Z.
- Program List Table Post Initialization (PLTPI) - The system requires initialization prior to being able to perform queuing operations. To automatically start the MQSeries system, the user may add the following programs to the CICS initialization PLT (PLTPI) list:

  MQPSENV

  MQPSTART

  Alternatives to this method can be found in “Initialization of the MQSeries System”, below.
- Program List Table Shut Down (PLTSD) - The MQSeries System needs to be shutdown prior to performing shutdown of CICS itself.
  Either:
  Place program MQPSTOP in the CICS shutdown PLT before the DFHDELIM statement.
  or
  Execute transaction MQST from a CICS terminal session.

Modify CICS start-up deck

- The MQSeries System (CICS) subsystem datasets must be allocated to the CICS partition. Member MQJLABEL.Z contains sample JCL, which must be added to the CICS start-up deck for this purpose.
- The sublibrary PRD2.MQSERIES must be added to the LIBDEF search chain for phases in the CICS start-up deck.

Recovery/Restart

Although the MQSeries System uses its own Recovery/Restart logic, it also uses the standard CICS file management. Therefore, it is important that all MQSeries VSAM clusters be defined in DFHFCT with the LOG = YES parameter. In addition the CICS logging facility must be activated (JCT = xx or YES).

If the above conditions are not fulfilled, unpredictable results may occur such as loss of messages or inaccurate values for message sequence numbers.

Uppercase translation

Queue Manager, queue and channel names are case sensitive on MQSeries Systems. If the MQSeries System on VSE has to send messages to other MQSeries Systems, the user *must* specify `UCTRAN = TRANID` or `UCTRAN = NO` in his CICS terminal definitions. If this is not done, the names entered from the MQSeries System panels will be translated into uppercase and may not match the actual names on the OS/2 or UNIX MQSeries System.

System setup

The following steps must be performed only once before the MQSeries System can be used.

1. The batch job **MQJSETUP.Z** must be executed successfully. This copies the member SYSIN.Z into a VSAM ESDS file.
2. Execute the MQSU transaction (Setup System Configuration File). The message "MQSERIES INSTALL COMPLETED" should be produced. If this message is not produced, then check the installation of the MQSeries product to make sure all the components are properly in place.

Initialization of the MQSeries System

There are three ways of initializing the MQSeries system.

– Starting transactions from a terminal (or by using CRLP on sequential terminals)
  Issue MQSE - (Setup Environment)

  There will be a delay of approximately one minute before the response INITIALIZATION COMPLETED is displayed.

  Issue MQIT

Warning: The first time MQSE is issued, a warning message will appear indicating that no system record has been defined. This is normal and will disappear once the system configuration record has been defined in MQMT.

– Using the MQSeries panels
  Issue MQSE - (Setup Environment)
  Issue MQMT (the main menu panel of MQSeries Administration displays)
  Select 2 - Operation
  Select 4 - Initialization/Shutdown
  Type I in the function field and press the Enter key.

– By defining entries in the CICS PLTPI table (refer to "Install CICS table entries" on page 8).

Note: If initialization is done before the System setup is performed a "MQ900000:MQSERIES VSE ENVIRONMENT not initialized" message will be produced. See the System Setup below.

Once the VSE/ESA environment has been established for the MQSeries System the following configuration entries must be completed:

1. Global System Definition must be defined.
2. At least one Local Queue definition.

Define global system definition

1. Review the sample "Global system definition" on page 68 and decide on desired values.
2. At the system prompt, type:

   ```
   MQMT
   ```
3. The Main Menu will appear. To select Configuration, type:

   ```
   1
   ```
4. The Configuration sub-menu will be displayed. To select the Global System Definition, type:

   ```
   1
   ```
5. Key in the desired entries and press PF6 to save the changes.

Other installation considerations

External Security - If external security packages are used (for example, ALERT**), please ensure that MQSeries System (CICS) LU6.2 sessions are 'signed - on' and authorized to execute channel driver transactions (that is, MQ01 and MQ03), and the message delivery transaction (MQ02).

In case of a sender channel, MQ02 is used as an outbound channel driver.

Note: Transaction names MQ01 and MQ03 may be modified to any naming conventions to fit the installation.

MQQA and MQQD are two Queue Maintain transactions which both point to the MQPQDEL program, used for updating queue records. MQQA is specifically for the Delete All function, while MQQD is for the Delete By Date/Time and the Reset Deleted Records By Date/Time functions. This allows the security package to prevent unauthorized use of these functions when started by the Queue Maintenance Operation Master Terminal task.

The CICS Journal Control Table may 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" will occur. This will be reflected in either the MQM System Log or the CSMT TD queue when a MQPUT call is executed trying to perform this function.

MQSeries System installation verification test

The installation verification test will use one local queue, the sample transaction TST2 and the program TTPTST2 provided with the release.

Note: Before running the installation test, the Global System Definitions must be completed (refer to "Global system definition" on page 68).

To configure the MQSeries System for this test, you need only create the queue using the MQMT administration screens, as follows:

▼ *Verifying the MQSeries System installation*

1. At the system prompt, type:

   ```
   MQMT
   ```

2. The Main Menu will appear. To select Configuration, type:

   ```
   1
   ```

3. The Configuration Sub-Menu will appear. To select Queue Definitions, type:

   ```
   2
   ```

4. The Define Queue Name screen will appear. Fill in the following:

   Object Type: **L**

   Object Name: **ANYQ**

   Press PF5 (Add Queue)

5. The Create Local Queue screen will then appear with default values. Press PF10 to bring up the extended screen and fill in the following:

   Usage mode: **N (Normal)**

   Physical File Name: **MQFI001 (file name from FCT)**

   Maximum Q Depth: **100**

   Maximum Message Length: **4096**[1]

   All other fields can remain with default values.

   To save the entry, press PF5

1. This size cannot exceed the maximum message size defined in the system definition and is about 750 bytes less than the maximum VSAM recordsize, however, the user may enter any larger number and then downsize to the suggested value provided by MQM.

6. Press PF2 to return to the Queue Main Options Screen.
7. To Display your Queue Definition, press:

   PF9
8. A selection screen will appear, use the cursor keys to select the queue, press any character and:

   <Enter>
9. A screen will display the queue parameters just entered. Visually verify that the correct data has been entered.

Congratulations. You have created your first MQSeries System queue.

You have now created a test configuration which will allow TST2 transaction to send messages to **ANYQ**.

Now we will attempt to send and receive messages locally.

The local installation verification test consists of five logical steps. First, you will initialize the MQSeries System runtime environment. Second, you will use the test program TTPTST2 to send a number of messages. Third, you will use MQMT to verify that these messages are on the queue. Fourth, you will use the test program TTPTST2 to read the messages. Finally, you will again use MQMT to verify that the messages were delivered.

Program TPTTST2 is activated when the transaction TST2 is typed. Parameters have to be specified according to the following format:

```
TST2 XXXX NN QQQQ Y
```

One or more spaces separate the parameters. Data entered is case sensitive.

XXXX	3 or 4 character function code required (as per help screen).
NN	Optional 1 or 2 character numeric field giving the number of messages to be processed, depending on the function code.
QQQQ	A field of up to 48-characters, giving the name of a queue.
Y	(optional) If timestamping is required.

10. Return to the Main Menu.
11. Select the Operation sub-menu.
12. Select Initialization/Shutdown of System.
13. On the Initialization/Shutdown screen, enter an `X` and press PF6 to shutdown the system. Then enter an I and press PF6 to initialize the system.

    This step reloads queue definitions, including the queue just created, into the runtime environment, and will take approximately one minute to execute.
14. Return to the Main Menu.
15. From the Main Menu, select the Monitoring option.
16. From the Monitor Menu, select Monitor Queue.
17. The Queue Monitor screen will appear showing **ANYQ** as the only defined queue. Note the number of messages currently on queue (Q DEPTH).
18. Move to another terminal, or to another workstation or window.
19. At the CICS prompt, type:

    ```
    TST2 PUT  10 ANYQ
    ```

    If you type TST2 with no parameters, the HELP screen for TTPTST2 usage will be displayed.
20. TTPTST2 sends the specified messages addressed to **ANYQ**.

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

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

where
nn = the number of messages you specified (10) and
hh:mm:ss = the time taken to process nn messages.

21. Return to the window (or workstation) running the MQMT Monitor Queue.
22. Press <**Enter**>. The number of messages for queue **ANYQ** should now equal nn, where nn = the value specified in Step 20.
23. At the CICS prompt, type:

    ```
    TST2 GET  10 ANYQ
    ```

    See note in step 19 above.
24. TTPTST2 reads the specified messages from **ANYQ**. You will receive the following message after successful completion of the transaction:

    ```
    FULL CYCLE HAS BEEN PERFORMED SUCCESSFULLY
      QUEUE USED - ANYQ
      NUMBER OF MESSAGES PROCESSED -  10
      TOTAL SECONDS..................... -  hh:mm:ss
    ```
25. Return to the window (or workstation) running the MQMT Monitor Queue function.
26. Press <**Enter**>. The Monitor Queue screen will still be showing **ANYQ** as the only defined queue. Note the number of messages on queue now (QDepth).
27. The screen will show the number on queue **ANYQ** to have decreased to zero, and the total number of messages read from the queue (LR) will have increased by the number you read using TTPTST2.
28. Exit MQMT.

You have now completed a *local* installation verification test demonstrating that two applications can send/receive messages via an MQSeries System queue. Realize that this test has not verified any communications links connecting you to a remote system.

There is a log queue with a default name of SYSTEM.LOG as indicated in the Global System Definition (Option 1 or 4 in the Configuration Main Menu) that is very helpful for debugging purposes.

29. Define queue name as SYSTEM.LOG using physical file name MQFLOG (file name from FCT) with Maximum Queue Depth 1,000,000 (please refer to steps 4 and 5 for details about defining a queue).
30. After the test, the log queue may be browsed by choosing the Browse Queue Records (Option 4) in the Master Terminal Main Menu, entering SYSTEM.LOG as object name with a proper QSN number.

Notes:

1. In order to expand this test to include a remote link, three steps are required.
   a. Install the prerequisite hardware and software required to support the selected transport protocol. Refer to the manufacturers directions for this installation.
   b. Define the desired MQSeries System channel(s). Refer to Chapter 6, “System operation”, on page 65, and coordinate with the remote system administrator to accomplish this.
   c. Configure the transmission queue(s) and remote queue(s) required for the MQSeries System to communicate over the channel.
2. In order for new queue definitions and channels to take effect at run time, the MQSeries System must be shut down by first closing channel, stopping the queues, and then shutting the System down. Then you must reinitialize the MQSeries System as in Step 13 above.

The MQSeries System software has now been installed and locally verified using the provided test programs. The administrative programs and the MQI libraries may be used now. But, before user applications may effectively use the system for message transmission, the MQSeries System must be fully configured.

This last step is the most critical step of the installation. It is expanded into the following three chapters. Chapter 3, “Planning”, on page 15, summarizes the planning for new installations. Chapter 4, “Configuration”, on page 23, provides the configuration guidelines. Chapter 6, “System operation”, on page 65, describes the MQSeries System administration screens used in the configuration.

Product information file

The Product Information File named PRODINFO.Z is located in the MQSeries sublibrary, and contains the software level and service history of the product.

Chapter 3. Planning

This chapter provides an overview of the considerations for implementing a distributed application using the MQSeries System. This chapter will present an overall framework for the planning of a distributed application and will expand on areas specific to MQSeries System.

A planning framework for distributed applications

As the term “middleware” suggests, the MQSeries System supports the creation of message-enabled applications, and resides between distributed applications and the underlying communications network. As such, it is imbedded in an often long process of planning and implementation.

Several disciplines are involved in this planning. These may be administered independently, resulting in separate but related planning domains for applications, systems, networks, etc., or, they may be integrated to a higher level of planning for the distributed environment. In either case, planning and implementation procedures will vary substantially from one organization to another. Yet, it is often desirable to have a frame of reference when discussing individual planning activities. It is for this purpose that a generic Distributed Planning Procedure is outlined below.

Tasks and responsibilities

Figure 1, on page 16 identifies tasks and allocates them to the individual or organization typically responsible. In the paragraphs that follow, each of the individual tasks is summarized. Those that include the MQSeries System are expanded further.

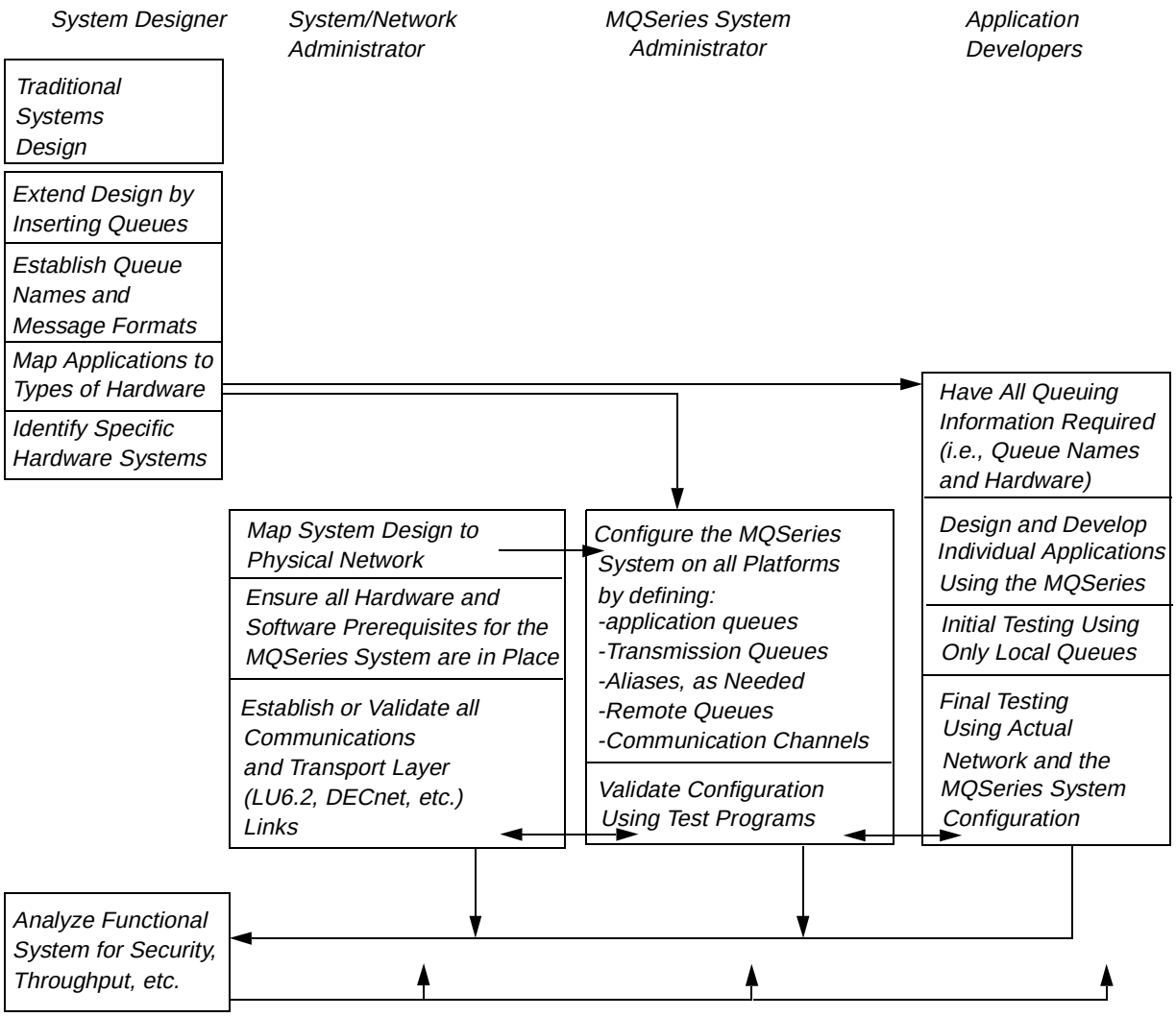


Figure 1. Tasks and responsibilities

System designer tasks

Traditional analysis and design

For new development, the design begins normally. Several well recognized methodologies exist for approaching the basic system design effort. Any one of these results in a functional decomposition of the overall system into a mesh network of processes depicting the flow of information through the designed system. The mesh may be arbitrarily complex based on the system requirements, but each process will be defined in terms of its local function and in terms of *data formats* exchanged with other processes.

For example:

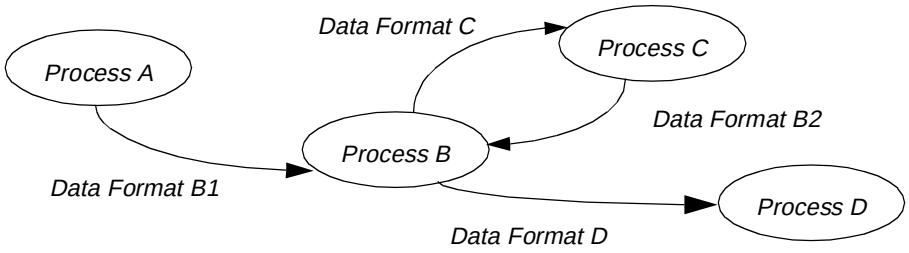


Figure 2. Typical data flow diagram

For existing systems which are to be modified to operate in a distributed environment, the above process may already be complete, or may have never been performed. If documentation at this level is not available, it must be created. The functional decomposition must be accomplished at least to the level that will identify each process which is a candidate for relocation. Each of the processes must be understood in terms of the *data formats* exchanged with other processes.

Extending to a distributed design

In order to extend a “traditional” design to a distributed environment, using messaging and queuing, there are a few essential steps.

- ***Identify which application processes are to be distributed.*** This might apply to all component processes or a subset of the entire system. In many cases, this is a simple step since the primary system goal will have been stated in terms of a desire to distribute a particular function.
- ***Isolate each such process by inserting queues*** (in the design) between it and the remainder of the system.
- ***Assign names to each of the required queues***. These names are the logical names which will be used by applications throughout the distributed environment to address the queues. It is convenient to think of the queue name as a *logical destination address* for a message. So, the names should be associated with the process which will *receive* messages via that queue.
- ***Define message formats for the new queues*** to replace the exchanged *data formats* in the original design. For example, notice the queues isolating **Process C** in the diagram below:

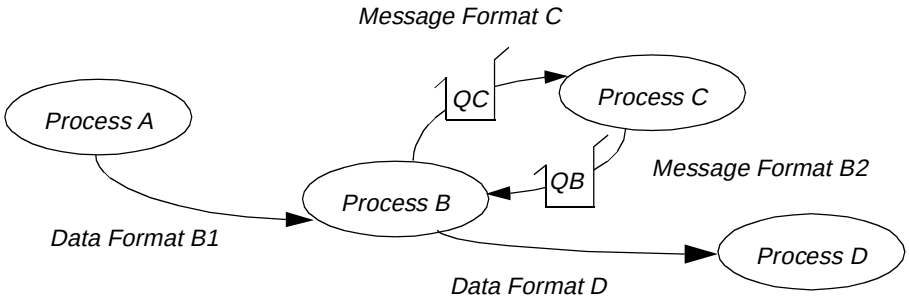


Figure 3. Data flow with queues

Mapping the design to the physical world

From a purist perspective, the highest level distributed design is complete. However, before much work can be done beyond the design, it must be mapped onto the physical distributed environment. This occurs in two steps.

Map the component processes (applications) to the "type" of host hardware on which they will be implemented. Such as Mainframe, VAX, UNIX, etc.

At this point, all information is known that is required by the application developers to begin development of individual applications. For more information see "Application developer tasks" on page 19.

- The formats of messages to be exchanged via the above queues.

The developers have no need for further knowledge of the underlying network or of the MQSeries System configuration. They specifically have no need to know where destination queues will eventually reside.

Map the component processes (applications) to the "specific" host hardware on which they will be operational. Such as: the Mainframe in Chicago, the VAX called "Mickey" in Engineering, the UNIX system at IP address 255.25.2.5, etc.

The naming conventions for these systems will be different for every company. In any case, whether formal or informal, these host systems will already be known to the enterprise network or they will have to be added to the network.

This step constitutes the last step of the distributed design and is the highest level map of the logical design to the physical network.

System / network administrator tasks

Map the logical design to the physical network

This is the detailed extension of the last design step. Verify or complete the map to specific hardware systems. (Completion of the map may be particularly necessary in the case of LAN implementations. The System Designer may not have low level LAN configuration knowledge, such as which workstations are served by which file servers.)

Ensure that hardware and software are in place

Verify that the hardware and software prerequisites for the MQSeries System are installed at each system involved in the distributed implementation.

Establish the transport layer of the network

This is a critical step which requires detailed system/networking knowledge of each platform but very little knowledge of the MQSeries System. This includes:

- Verify that physical communications links (paths through the network) exist between each of these systems.
- Establish any required transport layer definitions (LUs, PUs, NCP Gens, etc.) which are needed to support a logical point-to-point connection between the MQSeries Systems.

Some interaction with the MQSeries System administrator is required to complete the above steps. The information shared between the Systems/Network Administrator(s) and the MQSeries System Administrator(s) include:

- End points of point-to-point logical links.
- Number of links between systems.
- Transport protocol used.
- Transport specific names (LU Names, XIDs, Node Names, etc.).

MQSeries System administrator tasks

The MQSeries System Administrator is the focal point for a successful implementation since this is the one activity which touches all others. Interaction will be required with the System Designers, the Network Administrators, and the Application Developers.

The Administrator will have certain operational responsibilities after the distributed system has been implemented, but ***by far the most significant duty is the initial configuration of the MQSeries System queues***. This is the critical function which ties together the underlying transport network and the distributed applications. Briefly, this includes:

- Configuring the MQSeries System Message Queue Manager
- Configuring MQSeries System Local Queues
- Configuring MQSeries System Transmission Queues
- Configuring MQSeries System Queue Aliases
- Configuring MQSeries System Remote Queue Definitions
- Configuring MQSeries System Communications Channels

From a planning perspective, it should be realized that the queue configuration will require some level of coordination throughout the network, but will be accomplished on each individual system. Further, it should be recognized that configurations are typically built in three phases. These phases correspond to:

- ***Test configuration(s)*** to allow local testing of applications using queues, or initial testing of communications lines.
- ***Functional configuration*** to include all communications channels and all queues. This configuration allows full application functional testing. It has not been optimized for performance or been modified for any security or other installation specific requirements.
- ***Operational configuration*** which is an extension of the above after considering performance requirements, security requirements, etc.
- Details of all configuration activities are provided in Chapter 4, “Configuration”, on page 23, of this document.

Application developer tasks

Application development on individual platforms can begin relatively early in the implementation process. It can start as soon as the developers know:

- The platform on which the application will run and therefore, the MQSeries implementation specifics for that platform.
- The queue(s) on which the application will receive messages.
- The queue(s) to which the application will send messages (that is, the queues on which destination applications will receive messages).
- The formats of messages to be exchanged via the above queues.

The developers have no need for further knowledge of the underlying network or of the MQSeries System configuration. They specifically have no need to know where destination queues will eventually reside.

Development will proceed very much like traditional applications development. The only difference is the use of the MQI to interface to queues.

The MQI and other Application design considerations are described in detail in Chapter 7, “Application programming interface”, on page 105, of this document.

Including legacy applications in distributed designs

Legacy applications are commonly old, not well understood, not well documented, but they work. So, no one wants to touch them.

Such applications present an obvious paradox when they form a critical piece of a system that is to become distributed. *Their input/output interfaces cannot be readily altered yet they must be modified to support messages and queues.*

The solution is simple. The Legacy application is “sandwiched” between a *preprocess* and a *post-process* application which convert queues and message formats to/from existing data formats used by the legacy software.

To illustrate this, consider a segment of our earlier flow diagram:

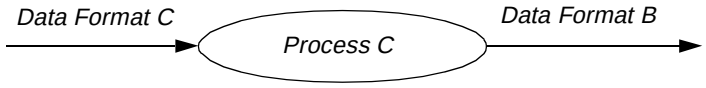


Figure 4. No messaging and queuing

If **Process C** can be directly modified to use the MQI to take advantage of messaging and queuing, then the result appears as:

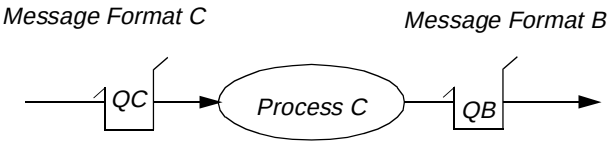


Figure 5. Messaging and queuing

But, if **Process C** is a legacy application which cannot be directly modified to use the MQI, then new *Pre* and *Post* processor applications are required, yielding:

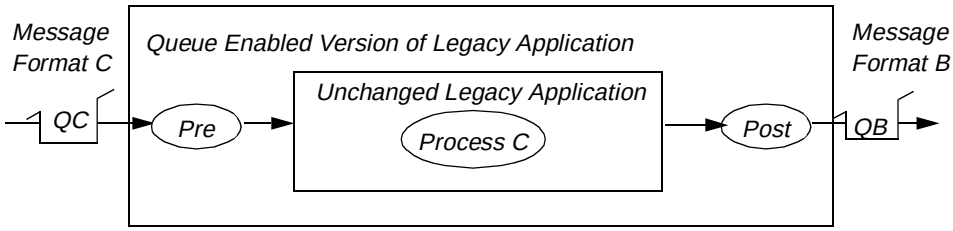


Figure 6. Queue enabled version of Legacy application

Of course, the *Pre* and *Post* processors must reside on the same hardware platform as the legacy application.

Planning considerations for VSE/ESA systems

Many of the following are detailed elsewhere in this manual. They are summarized here for convenience.

- *Prerequisite Hardware and Software* is defined in Chapter 2, “Installation”, on page 3.
- *MQI Features* have been implemented in a slightly different manner for different operating system environments. Chapter 7, “Application programming interface”, on page 105, should be reviewed closely to ensure that any features of particular interest are fully available on VSE/ESA.
- *Transport protocols* supported on VSE/ESA include LU6.2 only. MQSeries System implementations on other platforms may support additional Transport protocols (for example, TCP/IP on the RISC System/6000 and DECnet** on the VAX/VMS) but these are not currently available on VSE/ESA.

- *The Application Programming Language* supported with VSE/ESA is COBOL for VSE.
- Other languages may be usable at the application level, provided the customer constructs the API interface calls correctly (see Chapter 7, “Application programming interface”, on page 105).
- *Security* for MQSeries System queues, communication channels, and administrative programs can be established by taking advantage of native CICS security features. For more information, refer to the IBM CICS System Destination Guide.
- *Syncpoint participation* is supported on the IBM MQSeries for VSE/ESA. See Chapter 7, “Application programming interface”, on page 105.
- *The maximum message size* that can be written to a queue is affected by the Journal Control Table (that is, BUFSIZE). If this setting is too small, a CICS abend code of AFCL will be produced on a `MQPUT` Call.
- If recoverability is very important, then see “Dual Queue Support” on page 40. It outlines how this facility works. Essentially, a backup queue can be created at the same time the primary queue is updated. The MQSeries System handles recovery of this backup queue if it becomes out-of-sync with the primary queue.
- Queues can be fully rebuilt using a background MQPDUMP facility. This facility will copy all non-processed messages to a sequential file. A VSAM delete/define may be performed, to be followed by a reproduction of this sequential file back over the cleaned VSAM file. See the “Background batch modules” on page 98 for more information.

Chapter 4. Configuration

The MQSeries System software has now been installed on your system by following the instructions in Chapter 2, “Installation”, on page 3. However, it cannot converse with other MQSeries System installations or even perform local messaging until it is *configured*.

Building an effective configuration for MQSeries System operation is by far the most critical task to insure a successful implementation.

This chapter will explain the *concepts* required to properly configure the MQSeries System.

The mechanics of entering configuration data are detailed in Chapter 6, “System operation”, on page 65, as part of the Operator Screens Reference.

MQSeries System configuration elements

Configuring the MQSeries System requires that the administrator/operator define the following MQSeries System elements:

- Message Queue Manager
- Local Queues
- Transmission Queues
- Communications Channels
- Queue Aliases and/or Remote Queue Definitions

A clear understanding of each of these elements and an understanding of the MQSeries System message routing is needed in order to properly configure the system.

Note: The Configuration of these elements does not affect the Routine Environment until an initialization of the system is performed.

Queue names and message routing

Queue names are used in all MQI calls to identify the queue with which you want to work. Queue names are also included in message headers and, as the message traverses the network, are the basis for *routing* the message.

A fully qualified queue name consists of two parts:

- The *queue_manager_name*, which identifies an MQSeries System.
- The *queue_name*, which identifies the queue itself.

The full name may be written *queue_name@queue_manager_name*. If the queue_manager_name is omitted, it is assumed to be the local_queue_manager.

This two-part naming convention represents the essence of message routing for the MQSeries System. The fundamental routing algorithm is very simple:

The *queue_manager_name* identifies the MQSeries System on which the queue called queue_name resides.

A goal of IBM MQSeries is to hide the network details from the application. The application should not have to identify the system on which a particular queue resides. For this reason, applications can use *aliases* and *remote queue definitions*, which are explained later in this section.

Note: Some other operating systems, which the MQSeries System for VSE user may be communicating with, may be case sensitive. It is important to read “Uppercase translation”, on page 9, before devising a name for a queue, channel or Queue Manager.

Queue name format

Each part of the queue name is contained in a 48-character field, with the local-name part appearing first.

The character set that can be used for the queue manager name and queue name is as follows:

- Uppercase A - Z
- Lowercase a - z
- Numerics 0 - 9
- Period (.)
- Forward slash (/)
- Underscore (_)
- Percent sign (%)

Note: Leading or embedded blanks are not allowed.

Queue names and queue manager names that are shorter than the full field width can be passed by an application program, either by padding to the right with blanks, or by using a null (X“00”) character after the last significant character of the name.

The null character and any characters to the right of it (which are ignored), are treated as blanks. There can be blanks between the last significant character of the name and the null character.

For example, a single null character in the first character position of the queue manager name field can be used to default to the connected queue manager. This method is convenient for C programs.

Either method (right padding or null character) can be used for names that are passed by the application across the interface, but all names (including process-object names) that are returned by the queue manager are always padded to the right with blanks.

Any structure to the names (for example, the use of the period or underscore) is not significant to the queue manager.

Note: Names starting “SYSTEM” are reserved for the queue manager-defined queues.

Message queue manager

It is most reasonable to think of the Message Queue Manager as the *domain* composed of:

1. The MQSeries System disk directory containing the messaging and queuing configuration database.
2. Any Messaging and Queuing software which operates by using this database.

The latter includes all the MQSeries System software components described in “Software components of the MQSeries System”, on page 2.

Local message queues

In Chapter 1, “Product description”, on page 1, a *local queue* was defined as any queue residing on the same message queuing system as the application. In the MQSeries System, a local queue corresponds directly to a physical disk file which holds messages.

The *queue_name* of a local queue is used by all programs to access the queue.

In most cases, one local queue must be created for each MQI application running on the system. This queue is used by the MQSeries System to store *inbound* messages destined for the target application. Put another way, the application *receives* messages via its associated local queue.

The required local queues are normally identified by the System Designer who has enterprise wide responsibility for distributed applications. The Designer typically associates a *queue name* with an application program. For example, the designer may prescribe the following relationships:

Application	Queue Name
Accounts Receivable	Accts_Receivable
Accounts Payable	Accts_Payable
Order Entry	Ops_Orders
Shipping	Ops_Shipping
Inventory	Ops_Inventory

Table 1. Application and queue name

The publication of a list, as above, establishes a naming convention by which all developers understand how to address messages to a particular destination application. For example, from the above list, any program wishing to send a message to the Order Entry application, uses the MQI call set to *put* a message to the queue named Ops_Orders. Similarly, the Order Entry application itself receives messages (sent by other programs) by using the MQI call set to *get* messages from the queue Ops_Orders.

Note: While the normal case is one local (input) queue per application, there are cases in which an application may require:

- **Multiple local queues** (for example, high priority traffic on a separate queue)
- **No local queues** (for example, programs that generate messages, but never receive)
- **A shared local queue** (for example, multiple processes all servicing the same high volume queue)

Transmission queues

A special case of a local queue which is used to hold messages to be transmitted to another system is called a *transmission queue*.

Since a transmission queue is a local queue, it also corresponds directly to a physical disk file which holds messages. Beyond that, a transmission queue is substantially different from a *normal* local queue.

Whereas a local normal queue holds inbound messages, a transmission queue holds *outbound* messages. Whereas a local queue holds messages for a single application, a transmission queue interleaves messages destined for several different applications residing on the same remote MQSeries System.

Note: A transmission queue is associated with a communications channel. The messages on a transmission queue are processed only by the MQSeries System's Message Channel Agent (MCA). Normal MQI applications cannot directly access a transmission queue.

In most cases, one transmission queue must be created for each *adjacent* MQSeries System in the network. In this context, *adjacent* means any system with which you have a point-to-point *logical* connection at the MQSeries System level. Since the physical topology of the underlying transport network is hidden by the MQSeries System, this may or may not correspond to a point-to-point physical connection. But, it may be conceptually easier to think of the connection in physical terms.

While the normal case is one transmission (output) queue per adjacent MQSeries System, there are cases in which an MQSeries System may require:

- **Multiple transmission queues to the same destination** (for example, for higher throughput)
- **No transmission queues** (for example, if an MQSeries System is to be used for only local interprocess communications. While rare in normal operation, this arrangement is often useful in test scenarios.)

Though the System Designer typically identifies required local queues, the designer may not identify all required transmission queues. If not, the MQSeries System administrator must compile:

- A list of applications running on the local system
- A list of destination queues to which the local applications send messages
- A list of remote MQSeries Systems on which these queues reside

From the above compilation, it should be simple to determine the required transmission queues. Further system or application information will be required to identify special cases requiring more than one transmission queue per connection.

Communications channels

In Chapter 1, “Product description”, on page 1, a channel was defined as a unidirectional point-to-point communications link between two MQSeries Systems. The MQSeries System channel parameters are defined by using the Channel Definition screen in the MQMT program, as detailed in Chapter 6, “System operation”, on page 65. Each channel has a number of characteristics:

- Unique twenty character channel name
- Unique message sequence numbers
- Communications parameters required by the transport layer

The MQMT Channel Definition Screen defines only the communications link parameters. The associated transmission queue must be defined separately.

To fully implement an MQSeries System channel, you must perform functions on each of two systems. For example, in order to connect LAN_A to MAINFRAME_B, you must:

On the MQSeries System installed on LAN_A:

- Use the MQMT Queue Definition Screen to define an outbound transmission queue called, for example, **MAINFRAME_B**.
- Use the MQMT Channel Definition Screen to define communications parameters for the channel named, for example, **CHANNEL_1**. In the channel definition, you will specifically identify **MAINFRAME_B** as the name of the transmission queue for this channel.
- Insure all transport layer hardware and software is properly installed.
- Activate the MCA to process the LAN_A end of **CHANNEL_1**.

On the MQSeries System installed on MAINFRAME_B:

- Insure all transport layer hardware and software is properly installed.
- Use the MQMT Channel Definition Screen to define communications parameters for the channel named **CHANNEL_1**. (On the input end of a channel, no transmission queue is involved.)
- Activate the MCA to process the MAINFRAME_B end of **CHANNEL_1**.

Note: The above sequence of actions has established a channel in one direction only, from LAN_A to MAINFRAME_B. The same steps must be performed to create another channel, if desired, to allow messages to flow in the opposite direction.

Remote queue definitions

A remote queue definition is simply alternative logical name which can be used to address an MQSeries System queue instead of using the actual queue_name. A single name is provided for use by an application which relieves the application of needing to know the location (*queue_manager_name*) of the destination queue.

These extensions to the use of direct *queue_names* exist solely to simplify the work of developers and to improve the flexibility/portability of distributed applications.

A remote queue definition is simply a logical name defined on the local system which identifies a queue physically resident on another system. The *queue_name* so defined, can be used by applications to address the queue, but the MQSeries System will realize the queue is elsewhere and direct the messages to the remote site.

Note: MQI does not support GET operations directed to a remote queue.

To define a remote queue, one does not supply the same fields as when defining a local queue (for example, no file name, or record size), but must supply both:

- the *queue_manager_name* of the remote system

and

- the *queue_name* of the actual physical queue on the remote system

Optionally, you may also identify a transmission queue (other than the default transmission queue) which is to be used to send messages to the remote system.

When defining a remote queue, each entered name is validated by the MQSeries System as follows:

queue_name	must be unique among all names defined locally.
queue_manager_name	must match a local transmit queue, or the optional alternative transmit queue must be supplied.
remote_queue_name	must match a definition on the remote system but this cannot be validated locally.
transmit_queue_name	must match a local transmit queue, if present.

This extended queue identity is not visible to an application on the local system. Local applications use only the *queue_name.*

Note: Some other operating systems, which the MQSeries System for VSE user may be communicating with, may be case sensitive. It is important to read “Uppercase translation”, on page 9, before devising a name for a queue, channel or Queue Manager.

Conversely, the *queue_name* used locally is not visible to the remote system. In its place, the fully qualified remote queue name (*remote_queue_name @ queue_manager_name*) is inserted in the message header before transmission.

Aliases

An *alias* is similar to a *remote queue definition* in that it is an alternative logical name which can be used to address an MQSeries System queue instead of using the actual *queue_name*. An alias, however, is simpler than a remote queue definition.

An alias provides a simple one-to-one name substitution capability. It associates an alternative (alias) name with an already defined queue.

By defining an alias, the MQSeries System administrator has the ability to redirect message traffic. For example, if an application was originally coded to write to a queue called FRED, but we now want the output to go to JOHN, the redirection can be accomplished by redefining FRED as an alias for JOHN rather than as a real local queue.

The MQSeries System supports two other types of aliases beyond the simple queue alias. There is a *manager_alias*, which is simply an alias associated with an already defined queue_manager_name. There is a *reply_to_alias* which is somewhat more complex and infrequently used. This last type of alias will be explained more fully in “Alias queues, remote queues, and routing”, on page 30.

Note: An alias can be defined for a local queue, a remote queue, or a queue manager.

MQSeries System message routing

MQSeries System message routing is not to be confused with lower level network routing. The MQSeries System is normally concerned only with fixed, point-to-point routing which is substantially simpler than dynamic, adaptive, multi-hop, network routing algorithms. However, the many options available can make MQSeries System routing somewhat complex.

The MQSeries System must be explicitly configured (that is, queues, channels, aliases, etc. must be defined) to insure the desired flow of messages through the network. To do this effectively, the MQSeries System routing algorithm must be understood.

To understand MQSeries System routing, we will look first at the basic routing algorithm, then at the MQSeries System routing table, and finally at effects on routing which can be generated through aliases and remote queue definitions.

Basic message routing

Early in this chapter, it was noted that the two-part MQSeries System queue names embodied the essence of message routing for the MQSeries System. The fundamental routing algorithm is very simple:

> **The *queue_manager_name* identifies the MQSeries System on which the queue called *queue_name* resides**

The basic algorithm may be expanded by following the flow of a typical message from one system to another, as follows:

1. At the originating system, a message is presented (PUT) to the MQI with the two-part destination *queue_name*.
2. The MQSeries System examines the destination *queue_manager_name* to see if it matches the *local_queue_manager_name*. Typically, it does not match, so the MQSeries System knows the message goes to another system.
3. In this case, the destination *queue_manager_name* **must** match a transmission queue defined on the originating system. This is the default transmission queue to reach the specified *queue_manager.*
4. The message is enqueued to this transmission queue.
5. The MQSeries System MCA on the originating (output) system GETs the message from the transmission queue and sends it over the link to the remote system. Notice that the output MCA utilizes no routing logic.

6. The MQSeries System MCA on the destination (input) system receives the message from the communications link (and invokes routing logic to determine what to do with it).
7. The MQSeries System examines the destination *queue_manager_name* to see if it matches the *local_queue_manager_name*. Typically, it does, so the MQSeries System knows the message belongs “here”.
8. The MQSeries System then examines the destination *queue_name.* In this case, the destination *queue_name* **must** match a local queue defined on the destination system.
9. The message is enqueued to this destination queue.
10. The destination application receives (GETs) the message via the MQI.

From this example, several **basic configuration principles** may be observed:

1. MQSeries System routing logic is exercised independently on each system. Therefore, each MQSeries System must be configured individually, but all configurations must be coordinated to be effective.
2. Any configuration must have defined one local queue for each inbound destination on the system. These will be used to receive incoming messages. The defined *queue_name* must match the *queue_name* applications will use in message headers.
3. Any configuration must have defined one transmission queue for each remote destination system. These will be used to transmit outbound messages. The defined *queue_name* must match the *queue_manager_name* applications will use in headers of outbound message.

The MQSeries System routing table

Before exploring the routing logic further, it is useful to understand the MQSeries System’s **Routing Table** which is used to resolve all queue references. Note that the table is described here as a logical entity and may not exactly correspond to the data structure on a particular system.

MQSeries System queue names are of the form ***queue_name @ queue_manager_name***, each half of which is 48 characters. The MQSeries System Routing Table, however, is keyed to a single 48-character string. This string is normally a *queue_name* but will be called *Object_Name* to avoid/reduce confusion in this discussion.

An entry must exist in the Routing Table for each of the following:

- All **LOCAL** queues (Type=Local, Usage=Normal)
- All **TRANSMISSION** queues (Type=Local, Usage=Transmission)
- Any desired definitions for **REMOTE** queues
- Any desired **ALIAS_Q** names for queues
- Any desired **ALIAS_M** names for queue_managers
- Any desired **ALIAS_R** names for reply_to_queues

The format of each Routing Table entry varies according to type. This is summarized in the chart below.

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
Required	**LOCAL**	----	----	----
Required	**TRANSMIT**	----	----	----
Required	**REMOTE**	Required	Required	Optional
Required	**ALIAS_Q**	Required	----	----
Required	**ALIAS_M**	----	Required	Optional
Required	**ALIAS_R**	Required	Required	----

Table 2. Routing Table Format

Alias queues, remote queues, and routing

What happens to routing, when alias queues and remote queues are introduced into the algorithm? This can be seen by considering various routing scenarios and examining the results of these scenarios under the MQSeries System routing logic. These sample routing cases will also further illustrate normal routing cases.

In these cases, the ***QName @ QMgrName*** shown as input indicate the “actual” nature of the input. The routing algorithm results will be as indicated for each case.

Note: The following is not intended to suggest application level pseudo-code, but only to explain what happens within MQSeries System routing.

1. Application attempts to operate on queue identified as

   Queue_Name @ Local_Queue_Manager

   Routing Process:

   Queue_Name has to match a Routing Table entry for Local or Remote queue .... or error.

2. Application attempts to operate on queue identified as

   Queue_Name @ (Blank_Queue_Manager)

   Routing Process:

   Queue_Name has to match a Routing Table entry for Local or Remote queue .... or error.

3. Application attempts to operate on queue identified as

   Queue_Name @ Remote_Queue_Manager

   Routing Process:

   Remote_Queue_Manager has to match a Routing Table entry for Transmit queue.... or error, *Queue_Name* is ignored.

4. Application attempts to operate on queue identified as

   Remote_Queue_Name @ (Blank_Queue_Manager)

   Routing Process:

   Remote_Queue_Name has to match a Routing Table entry for Remote queue .... or error.

5. Application attempts to operate on queue identified as

   Alias_Name @ Local_Queue_Manager

   Routing Process:

   Alias_Name has to match a Routing Table entry which resolves to another Routing Table entry for Local or REMOTE queue .... or error.

6. Application attempts to operate on queue identified as

   Alias_Name @ (Blank_Queue_Manager)

   Routing Process:

   Alias_Name has to match a Routing Table entry which resolves to another Routing Table entry for Local or REMOTE queue .... or error.

7. Application attempts to operate on queue identified as

   Alias_Name @ Remote_Queue_Manager

   Routing Process:

   Remote_Queue_Manager has to match a Routing Table entry for Transmit queue .... or error, *Alias_Name* is ignored.

8. Application attempts to operate on queue identified as

   Some_Queue_Name @ Alias_Queue_Manager

   Routing Process:

   Alias_Queue_Manager has to match a Routing Table entry with type ALIAS_M which resolves to *Local_Queue_Mgr_Name*. Second pass through search logic resolves *Some_Queue_Name* to either case (1) or (5) above.

   OR

   Alias_Queue_Manager has to match a Routing Table entry with type ALIAS_M which does NOT resolve to *Local_Queue_Mgr_Name.*

   This case is handled same as case (3) or (7) above. No second pass through the search logic is required. *Some_Queue_Name* is ignored.

   OR

   Error.

9. Message Channel Agent receives inbound message with destination queue identified as:

   Some_Queue_Name@Some_Queue_Manager

   Routing Process:

   Some_Queue_Manager has to match a *Local_Queue_Manager_Name* and *Some_Queue_Name* is resolved as in case (1) or (5) above with a single pass through the search logic.

   OR

   Some_Queue_Manager has to match a Routing Table entry with type Alias_M which resolves to the *Local_Queue_Manager_Name*.

   On second pass through search logic, *Some_Queue_Name* is resolved as in case (1) or (5) above.

   OR

   Some_Queue_Manager has to match a Routing Table entry with type Transmit and is handled same as case (3) or (7) above.

   Some_Queue_Name is ignored.

   OR

   Some_Queue_Manager is invalid.

Other alias types

In some cases, it is desirable to have multiple channels, and multiple transmit queues defined for the same remote destination system. This conflicts with the standard use of the queue_manager_name as the transmit queue name.

The extension of the REMOTE queue definition to include a TRANSMIT queue is convenient for most such cases, but may be undesirable at a central “server” system which must deal with a large number of remote systems. The server would require a large number of REMOTE queue definitions in order to handle anything more than one TRANSMIT queue per system.

A Routing Table entry type ALIAS_R provides a mechanism to allow the name for the response transmission queue to be expanded at the originating system.

This may be thought of as a “Reverse Queue Manager Alias” or as a “Response Class” or as a “Response Category”.

It is a relatively simple concept which simultaneously frees a remote server from the need to define a long list of REMOTE queues, and frees the local application from the need to know details of the transmit queue structures, and allows the local application code to be completely portable.

For example:

This example shows the use of reply aliases and manager aliases to reduce the definitions required at a central server site.

An application running on SYS1 originates a message to a remote server and specifies ***Reply_to_Queue*** = PRIORITY (and ***Reply_to_Queue_Manager*** = BLANK).

At SYS1, the Routing Table contains three related entries:

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
My_Queue	LOCAL	----	----	----
PRIORITY	ALIAS_R	My_Queue	SYS1_PRI	----
SYS1_PRI	ALIAS_M	----	SYS1	----

Table 3. Local routing table

During outbound processing, the MQSeries System finds that PRIORITY matches a Routing Table entry of type ALIAS_R, and substitutes MY_QUEUE @ SYS1_PRI into the outbound ***Reply_to_Queue*** fields.

At the remote server, the Routing Table contains one related entry:

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
SYS1_PRI	Transmit	----	----	----

Table 4. Remote server routing table

When the server has completed processing the original message, the response is queued to the transmit queue SYS1_PRI.

Back at SYS1, the response arrives with ***QMgrName***=SYS1_PRI. This is resolved through the Routing Table to match SYS1 and so the message is accepted and enqueued to the local queue MY_QUEUE.

Portable application code

Not only did the above result in minimizing Routing Table entries at the server, but also it promotes totally portable application code.

Consider the case in which the network in the above example is to be expanded by adding a new system called SYS2. The new system will run the same application software as SYS1, and will post requests to the same server application.

By simply copying the unmodified application (executable) code from SYS1 to SYS2, and making the following Routing Table updates, all will work correctly.

At SYS2, the Routing Table contains three related entries:

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
My_Queue	LOCAL	----	----	----
PRIORITY	ALIAS_R	My_Queue	SYS2_PRI	----
SYS2_PRI	ALIAS_M	----	SYS2	----

Table 5. Additional system routing table

At the remote server, the Routing Table expands by only one related entry:

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
SYS1_PRI	Transmit	----	----	----
SYS2_PRI	Transmit	----	----	----

Table 6. Remote server's new routing table

At both SYS1 and SYS2, the application code uses the name "PRIORITY" as the ***Reply_to_Queue.*** The ALIAS_R logic resolves this correctly via the Routing Table and correctly directs response traffic through the server to two different remote systems through TRANSMIT queues which are not the default queues.

This, of course, can be extended to any number of *Message_Queue_Managers* and to any number of TRANSMIT queues used for Response messages.

Recommended naming conventions

The naming (of queue_managers, queues, and aliases) used in the MQSeries System can be very flexible. Each organization will have its own view of how these names should be constructed. Beyond conforming to the format described in "Queue name format", on page 24, choosing names is left entirely in the hands of the user organization. However, a few suggestions are provided below.

1. **Do not use very long names:** Though the name fields are 48 characters long, very long names are cumbersome. Also, in some cases, the MQSeries System displays or messages may truncate very long names due to screen size limitations. In most cases names substantially shorter than 48 characters are sufficient.
2. **Attempt to configure the MQSeries System so that all queues may be referred to by a one-part name:** This will maximize the "network independence", or minimize the network topology knowledge required, of the distributed applications. It is desirable for applications to use only a *queue_name* rather than the two-part *queue_name@queue_manager name* construct, allowing the MQSeries System, through its routing table, to determine the location of the queue. This can easily be accomplished by using remote queue definitions and/or aliases to identify all remote queues. (In installations which require access to a large number of remote queues, this may be too cumbersome to configure.)
3. **Use aliases:** First, this can avoid the need to change application source code when the network changes or when a remote application changes. Also, aliases can be used to resolve incompatibilities between different naming domains. For example, if two computers

in different companies are talking via the MQSeries Systems, each company will probably want to name their own queues and queue managers. A “territorial” dispute is not uncommon. Such conflicts can be resolved by using aliases to “translate” names at the border.

Configuration capacities

In the current release, the major configuration elements are limited as follows:

Queue Managers:	One
Local Queue Definitions:	Unlimited. However, the User may set a maximum number of allowed queues via the Global System Definition screen.
Alias Definitions:	Unlimited.
Remote Queue Definitions:	Unlimited.
Total Queue Objects:	Unlimited.
Object Handles:	Unlimited. However, the User may set a maximum number of allowed connections via the Global System Definition screen.
Channels:	Unlimited.
MCA Processes:	Unlimited.
Maximum Message Size:	32000 bytes (excluding 736 byte header). The User may set a system-wide maximum message size via the Global System Definition screen. The User may also set a maximum message size for each queue via the Queue Definition screen.

Configuration worksheets

The set of sample worksheets in Appendix F, “Configuration worksheets”, on page 277 is presented in a format intended for duplication and use by the MQSeries System administrator or other individuals who design, configure, or require knowledge of the MQSeries System network.

The worksheets presented are:

- System List (Message Queue Manager Names)
- Application List (Queue Names & Host Systems)
- Application Look at Queues
- System Look at Queues
- Channel List
- MQSeries System Configuration (Routing Table) Work Sheet

Each of the worksheets is presented one-worksheet-per-page in Appendix H. The purpose and field descriptions appear at the beginning of each worksheet. Users may use all, some, or none of these worksheets at their discretion.

Configuration examples

Four sample configurations are presented below.

Simple network - minimum configuration

Consider two systems, one in Chicago, one in Boston. Each system has a single Message_Queue_Manager which has the same name as the host city.

Both Chicago and Boston run copies of the same two applications, **Application_1** and **Application_2**, which are served by local queues **App_1** and **App_2** respectively.

Any application must be able to talk to any other application, but no segregation of traffic is required on the transmission between nodes. So, the default transmission queues are sufficient.

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_1	Local	---	---	---
App_2	Local	---	---	---
Chicago	Transmit	---	---	---

Table 7. Minimal Boston routing table

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_1	Local	---	---	---
App_2	Local	---	---	---
Boston	Transmit	---	---	---

Table 8. Minimal Chicago routing table

With the above configuration, applications at Boston may put messages to:

App_1 (The LOCAL application)

or

App_1 @ Chicago (The REMOTE application)

or

App_2 (The LOCAL application)

or

App_2 @ Chicago (The REMOTE application)

Similarly, applications at Chicago may put messages to:

App_1 (The LOCAL application)

or

App_1 @ Boston (The REMOTE application)

or

App_2 (The LOCAL application)

or

App_2 @ Boston (The REMOTE application)

Simple network - improved configuration

This simple configuration in the preceding example is workable, but it requires the applications to be aware of “Boston” and “Chicago” as the existing transmission queues.

This configuration could be improved as follows:

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_1	Local	---	---	---
App_2	Local	---	---	---
Rem_App_1	Remote	App_1	Chicago	---
Rem_App_2	Remote	App_2	Chicago	---
Chicago	Transmit	---	---	---

Table 9. Improved Boston routing table

With the above configuration, and complimentary changes to the Chicago Routing Table, applications at either Chicago or Boston may put messages to:

App_1 (The LOCAL application)

or

Rem_App_1 (The REMOTE application)

or

App_2 (The LOCAL application)

or

Rem_App_2 (The REMOTE application)

Simple network - improved configuration #2

A similar result could also be achieved with an alternative Routing Table using ALIAS_M entries, for example:

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_1	Local	---	---	---
App_2	Local	---	---	---
Remote	Alias_M	---	Chicago	---
Chicago	Transmit	---	---	---

Table 10. Improved Boston routing table using ALIAS_M

With the above configuration, and similar changes to the Chicago Routing Table, applications at either Chicago or Boston may put messages to:

App_1 (The LOCAL application)

or

App_1 @ Remote (The REMOTE application)

or

App_2 (The LOCAL application)

or

App_2 @ Remote (The REMOTE application)

Complex network - recommended configuration

Consider three host systems, one in Chicago, one in New York, one in Boston. Each of these systems has a single Message_Queue_Manager which has the same name as the host city.

Both Chicago and Boston run copies of the same four applications, **Application_1**, **Application_2**, **Application_3**, and **Security**. At both locations, these applications are served by local queues **App_1**, **App_2**, **App_3,** and **Sec** respectively. The first three applications at these sites interact only with a server at New York but not with each other. **App_3** uses a segregated priority transmission queues to and from the server.

New York is a centralized server site running two applications, **Server** and **Security**. **Server** is an “advanced” application which is served by two local queues **Nor_Req** and **Pri_Req**. Typically the remote applications **#1** and **#2** send normal traffic to **Nor_Req**. Application **#3** sends “high priority requests” to **Pri_Req**.

At all three locations, the **Security** applications may talk to any other **Security** application but their “classified” traffic must be segregated from the other applications' traffic. That is they must have a separate transmission queue.

Finally, in addition to these 3 host systems, there are fifty (50) distributed LANs, one in every state. Each LAN supports up to 20 applications which can generate both normal and priority requests to **Server** at New York. The normal and priority traffic must have segregated transmission queues to and from the server system.

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_1	Local	---	---	---
App_2	Local	---	---	---
App_3	Local	---	---	---
Sec	Local	---	---	---
NewYork	Transmit	---	---	---
NY_Priority	Transmit	---	---	---
NY_Secure	Transmit	---	---	---
Chicago	Transmit	---	---	---
Chi_Secure	Transmit	---	---	---
Sec_NY	Remote	Security	NewYork	NY_Secure
Sec_Chi	Remote	Security	Chicago	Chi_Secure
Nor_Req	Remote	Nor_Req	NewYork	---
Pri_Req	Remote	Pri_Req	NewYork	NY_Priority
Pri_Reply	Alias_R	App_3	Boston_Pri	---
Boston_Pri	Alias_M	---	Boston	---

Table 11. Boston host routing table

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_1	Local	---	---	---
App_2	Local	---	---	---

Table 12. Chicago host routing table

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
App_3	Local	---	---	---
Sec	Local	---	---	---
NewYork	Transmit	---	---	---
NY_Priority	Transmit	---	---	---
NY_Secure	Transmit	---	---	---
Boston	Transmit	---	---	---
Bos_Secure	Transmit	---	---	---
Sec_NY	Remote	Security	NewYork	NY_Secure
Sec_Bos	Remote	Security	Boston	Bos_Secure
Nor_Req	Remote	Nor_Req	NewYork	---
Pri_Req	Remote	Pri_Req	NewYork	NY_Priority
Pri_Reply	Alias_R	App_3	Chicago_Pri	---
Chicago_Pri	Alias_M	---	Chicago	---

Table 12. Chicago host routing table (Continued)

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
Nor_Req	Local	---	---	---
Pri_Req	Local	---	---	---
Sec	Local	---	---	---
Chicago	Transmit	---	---	---
Chicago_Pri	Transmit	---	---	---
Chicago_Sec	Transmit	---	---	---
Boston	Transmit	---	---	---
Boston_Pri	Transmit	---	---	---
Boston_Sec	Transmit	---	---	---
Alabama	Transmit	---	---	---
Alabama_Pri	Transmit	---	---	---
Repeat above pair for all 50 State LANs				
Wyoming	Transmit	---	---	---
Wyoming_Pri	Transmit	---	---	---
NY_Priority	Alias_M	---	NewYork	---

Table 13. New York host routing table

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
LAN_App_1	Local	---	---	---
Repeat above for all LAN Applications				
LAN_App_20	Local	---	---	---
NewYork	Transmit	---	---	---
NY_Priority	Transmit	---	---	---
Nor_Req	Remote	Nor_Req	NewYork	---
Pri_Req	Remote	Pri_Req	NewYork	NY_Priority
Nor_Reply_1	Alias_R	LAN_App_1	*StateName*	---
Pri_Reply_1	Alias_R	LAN_App_1	*StateName*_Pri	---
Repeat above two lines for all LAN Applications				
*StateName*_P ri	Alias_M	*StateName*	---	---

Table 14. State LAN routing table (identical at each site except for StateName)

With the above Routing Table configurations, applications at either Boston or Chicago or any of the 50 State LANs may PUT messages to:

Nor_Req (to Server at NewYork via “normal” path)

or

Pri_Req (to Server at NewYork via “fast” path including segregated Transmission Queue)

Any replies from **Server** may be specified by the originating application to be returned via **Pri_Reply** (or **Pri_Reply_n** for LAN Applications) which will use the segregated transmission queue for high-priority responses from New York back to whichever system originated the request.

Also, applications at the Boston or Chicago hosts may PUT messages to:

Sec_NY (to Security at NewYork via “secure” path including segregated Transmission Queue)

Notice that the **Security** applications are fully defined as REMOTE queues at all of the three major hosts. Thus no ALIAS_R routing entries are required, yet all traffic (including responses) can flow over the segregated Secure transmission queues.

Finally, consider the Routing Table entries required to support the **Server** application at NewYork. Entries are needed for 106 TRANSMIT queues (2 to each of 50 LANs and 3 to each of the other hosts). While this is a large number of entries, realize that it allows for segregated responses to each of more than 1,000 applications (20 at each LAN plus those at the hosts). To provide this same capability using only REMOTE queue definitions at the server (and not using ALIAS_R logic) would require the New York Routing Table to be over 2,000 entries (a normal path and a priority path to each remote application).

Dual Queue Support

A local queue can have a mirror image generated. This is done by filling the field “Dual Update Queue” when defining this queue. The local queue being mirrored will be called “Primary Queue” and the mirror queue will be called “Dual Queue”. This Dual Queue is also a local queue. The Dual queue has to be defined before completing the “Dual Update Queue” field.

When an application writes a message to the Primary queue, the message is also placed into the Dual queue. When a message is retrieved (logically deleted) from the Primary queue, the message is also (logically) deleted from the Dual queue.

Dual queuing has been implemented to increase the data integrity. Therefore, the following is strongly recommended :

- - Make the Primary queue unique in a VSAM cluster.
- - Make the Dual queue unique in a VSAM cluster.
- - Have these 2 clusters on different DASD volumes.

If the Primary queue is not empty when the Dual Update Queue field is updated via Queue Definition Dialogs, the Dual queue will be synchronized with the Primary queue as soon as the system is initialized (or these queues are closed then reopened). More generally, if for any reason the Dual queue does not match the Primary queue (or the Dual queue becomes unavailable), the queue is placed in recovery state. As soon as the Dual Queue becomes available again, a Queue Recovery task will make sure that the Dual queue is placed back in synch with the Primary queue.

If the Primary queue becomes unavailable, there is no automatic way to make the Dual queue take over. Instead, a manual intervention is needed and the batch program MQPUTIL is to be used. Therefore the following scenario is suggested:

1. Stop the MQSeries Queue Manager using MQMT dialogs.
2. Close MQSeries VSAM clusters by using CEMT, more precisely: the configuration file, and the 2 clusters containing the primary and Dual queues.
3. Backup the current configuration file.
4. Run the program MQPUTIL with the “DUALQ TAKEOVER command. (refer to“Background batch modules”, on page 98 for more details). This will make the Dual queue become the primary queue.
5. Reclaim space of already delivered messages and resequence by running job MQJREORG against the dual queue. This step is optional but recommended.
6. Reinitialize MQSeries by using MQMT. However, you should be aware that now you are working without a Dual queue anymore.
7. Try to understand the problem and resolve it. This may take a long time (hours if it was due to physical DASD I/O errors).
8. When it is repaired, and MQSeries has stopped, copy the current primary queue (former Dual queue) to the real primary queue (by using VSAM REPRO).
9. Restore the original Configuration file that you saved in step 3.
10. DELETE and REDEFINE clusters hosting Dual queues.
11. Restart MQSeries.

System configuration examples

For VTAM:

It is unlikely that the MQSeries System will require any changes to:

- VTAM start parameters and
- the definition of CICS Systems to VTAM

However, all involved LUs must be defined.

For CICS

Sample definitions for CICS tables may be found in the sublibrary PRD2.MQSERIES. However, other definitions are specific to the user environment and have to be done manually by using the CEDA transaction or DEFINE commands if using the DFHCSDUP batch program. This involves connection definitions and session definitions.

Connection definition:

CICS uses the connection name to identify the other system(s). For example, if sessions in VSE1 are to converse with sessions in VSE2 and MVS, the VSE2 connection must be defined in VSE 1, and vice versa. All involved sessions and terminals must also be defined:

VSE1	VSE2	MVS	Refer to
CONN(VSE2)	CONN(VSE1)	CONN(VSE1)	**Connection Definition**
CONN(MVS)			
Sessions	Sessions	Sessions	**Session Definition**
Terminals	Terminals	Terminals	**Terminal Definition**

Table 15. General Definition.

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

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

Table 16. Object Characteristics of Connection

The above settings along with default values are sufficient for operation. For other parameters, refer to CICS/VSE 2.3 Resource Definition Guide (SC33-0708).

Connection status can also be displayed by typing `CEMT I CONN:`:

STATUS: RESULTS - OVERTYPE TO MODIFY	
Conn(VSE2) Net(xxxxxxxx)	Ins Acq
Conn(MVS) Net(xxxxxxxx)	Ins Rel

Table 17. CEMT I CONN display output.

Session definition:

Type - `CEDA DEF SESSION G(MQSERIES)` - to create session names and fill in the fields with the following values::

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
Operational Properties	Autoconnect	Yes
	Buildchain	Yes
	RELreq	No
	Discreq	No
Recovery	RECOvoption	Sysdefault

Table 18. CEDA V SESS display parameter settings.

The above settings along with default values are sufficient for operation. For other parameters, refer to CICS/VSE 2.3 Resource Definition Guide (SC33-0708).

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

For the MQSeries System:

MQSeries channel definition

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:52:01                    Channel List                            VSE2
MQMMCHN                                                             QMMC

S    CHANNEL NAME     TYPE  STATUS LAST MSN  LAST CHECKPOINT

  C1                   R   ENABLE   000009       10:28:31
  MQM7.LU62.VSEF.DEVL  R   ENABLE   005054       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 7. MQSeries for VSE/ESA Channel List screen

IBM MQSeries for VSE/ESA product configuration guidelines

Background information

IBM MQSeries for VSE/ESA is a CICS application written in COBOL. It uses LU 6.2 protocol to exchange messages with peer MQSeries Systems. The user application (also a CICS application) interacts with the MQSeries product via MQI calls aided with CICS SYNCPOINT commands to keep resources synchronized. MQSeries messages are stored in queues with map to queue files. The queue file is a logical structure under VSAM and several queues can be defined to use a physical VSAM file. Set up parameters can affect performance significantly. For example, with 4KB (4096 bytes) page sizes, a queue file which has a message size greater than 4KB should be defined as SPANNED in the cluster of 4KB CI (control interval) size.

Note: There are performance implications when using VSAM SPANNED records. A Max RECORDSIZE of 16000 defined for the VSAM cluster should perform better than a CI size of 4k and SPANNED records because of the special handling for SPANNED records which includes:

- Checking the record type: first, middle, low
- buffer allocation requests

The I/O response time of a queue file request is subject to the contention of the I/O channel it accesses and the DASD volume in which it resides. A high activity transmission queue or the configuration file should not be placed in a high contention disk volume or disk bank.

Transaction program names are limited to 4 characters as the transaction IDs in CICS are limited to 4 characters.

The performance of the MQSeries System CICS transactions can also be influenced by the priorities set up for these transactions.

Since the MQSeries System can only allocate conversations after the LU6.2 sessions are bound, the local LU with its partner LU must be defined to VTAM and CICS.

General

All the following guidelines refer to the MQMT administration dialogs.

There are three levels of configurations:
- the Queue Manager Configuration
- the Channel Configuration
- the Queue Configuration

Some fields are the same in all three levels, for example, the Maximum Message Size. The size defined in the Channel Configuration must be equal to or greater than the largest message size that is accessing this channel. The size defined in Queue Manager must also be the largest of all of those defined in the channels under this Queue Manager. Each level of Maximum Message Size utilize 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 68), use the following guidelines:

Checkpoint Global Timer: The time interval (in seconds) set to take a global checkpoint of all active channel parameters. A value of 60 seconds is sufficient since it provides a good recovery period and utilizes a reasonable amount of resources.

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, such as 200.

Maximum Concurrent Queues: The maximum number (integer) of simultaneous open LOCAL queues allowed under this Queue Manager. One may neglect the overhead and set it to a large number, such as 200.

System Wait Interval: The maximum polling time (in seconds) for the system monitor program after the system initiation. Normally, a value of 5 seconds should suffice since it has been found to be good practice to drain the incoming message queue before sending out any pending messages.

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. This task starts up the trigger program and schedules the processes which reclaim resources held by abnormally terminated applications. If there are too many, the System Wait Interval should be reduced to schedule this clean-up process more frequently.

Maximum Q Depth: The Maximum number of active messages per queue (integer) allowed by the Queue Manager. 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 will be the limiting value for the queue. The value should be set to the maximum number of messages expected to be queued before any application starts to process them. Adding an extra 100% as a safety factor should be sufficient.

Maximum Message Size: The maximum number of characters per message (integer) that is allowed for this Queue Manager. This field needs to be large enough to accommodate the largest message. For example, if the anticipated largest message is 10 KB (10,240 bytes), this field should be set to 10240. Although one may set 32 KB to meet a 10 KB requirement, some valuable resources will be wasted.

Note: Since messages are actually stored in VSAM clusters, the maximum message size has to be calculated using the largest RECORDSIZE among all VSAM clusters containing queues. Each record is prefixed by a Message Header of 736 bytes for identification and description. Therefore, if for instance the largest CI is 4K (4096 bytes), the maximum RECORDSIZE is 4096-7=4089 bytes, and the Maximum Message size might be: 4089-736 = 3353 bytes.

Maximum Single Q Access: This field defines the maximum number of MQOPEN calls (integer) against any queue handled by this Queue Manager. A value of 1000 calls would be a good cushion while consuming little overhead, if the maximum number of opens for each queue in the system is 100 calls.

Maximum Global Locks: The maximum number of entries (integer) that the Queue Manager may use to maintain uncommitted MQPUT/MQGET calls per queue for the system for recovery. In practice, a value of 500 is normally used.

Maximum Local Locks: The maximum number of entries (integer) that the Queue Manager may use to maintain uncommitted MQPUT/MQGET calls per queue per task for recovery. Since an entry of a Local Lock is deleted once the application issues an explicit SYNCPOINT CICS command to commit updates, the more often an application takes the checkpoint, the less the Maximum Number of Local Locks is needed. A value greater than the largest of the maximum messages per batch of all channel records should be specified. A value of 20 is usually sufficient.

Checkpoint Threshold: The Maximum number of queue accesses (integer) between checkpoints to be taken by the Queue Manager. The smaller the value specified, the more often the Queue Manager will take 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 less resources will be consumed. A value in the range of 100 to 1000 is a good compromise between the performance and recovery speed.

Channel configuration guidelines:

When configuring the channel (see “Channel definitions”, on page 80), use the following guidelines:

Allocation Retries:

Number of Retries: The retry count field represents the number of times (integer) an allocation is retried when the conversation has not been established. A retry count of less than 10 times should be sufficient, as if this value is exceeded, the system may under stressed and further retries at this time could be counterproductive.

Note: When configuring a new environment, failures occurring more frequently than this may indicate a network problem. An investigation of the problem LU and its associated resources should be conducted to ensure the session is bound and to establish why the conversation cannot be allocated.

Delay Time-Fast: The time interval (in seconds) that allocation of conversation will be retried for the first cycle of retries. A value of 1 to 5 seconds is enough for this field. The time interval of 5 seconds would be used for a slow environment, such as a dial-up SDLC.

Delay Time-Slow: The time interval (in seconds) that allocation of conversation will be retried for the next cycle of retries should the first cycle of retries fails. A size between 3 and 10 seconds should be sufficient, 3 seconds for a normal environment and 10 seconds for a slow environment.

Get Retries:

Number of Retries: The number of MQGET retries (integer) when queue is depleted. If a transmission queue is empty, Queue Manager will retry at the Delay Time interval before disconnecting the channel or making a request to disconnect the channel.

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 “best value” may vary from 1 to 20 seconds. The longer the Delay Time is specified, the less frequently a channel is reopened. For time-consuming dial-up connections, a value of 20 seconds would be reasonable.

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 to set a simple disconnect interval similar to that provided on other MQSeries platforms.

Max Messages per Batch: Only one message per batch is supported.

Message Sequence Wrap: The MSN Wrap count (integer) represents the highest Message Sequence Number (MSN) value which will be used on this channel, after which the MSN will revert to 1. The value of the MSN Wrap count must be the same at both the sending and receiving ends of the channel. 999999 is the recommended value.

Max Transmission Size:

The mutually accepted maximum number of characters per transmission (integer). Since MQSeries for VSE/ESA does not support the use of segmented messages, this value should be at least 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 per messages (integer) that is allowed for this channel. It might be up to the maximum value for the RECORDSIZE of the VSAM cluster in which the transmission queue is defined minus the message header (736 bytes). For instance, provided a VSAM CI of 4K, the maximum RECORDSIZE is 4096-7=4089 bytes, and the Maximum Message size might be 4089-736 = 3353 bytes.

Connection ID: A four-byte field (character) identifying the connection required by the sender, optional for the receiver.

TP Name: The remote task ID (character) of the receiver on a remote CICS region or a Transaction Program name on a remote system. Required by the sender. Since CICS uses four bytes as the transaction Id, for CICS to CICS conversation, only the first four bytes of remote task ID are meaningful.

Note: VSE will convert the name to uppercase. The corresponding name on the remote system should be defined as all uppercase.

Checkpoint Frequency: A checkpoint event of this channel will be taken after the specified I/O activities have occurred. The “best value” varies from 10 to 1000 (in seconds) depending on emphasis of the system throughput versus the channel recoverability.

Checkpoint Time Span: A checkpoint event of this channel will be taken after the specified time interval (in seconds) has expired. A value of 10 seconds will not present too much overhead.

Queue configuration guidelines:

When configuring the Queue (see “Queue definitions”, on page 70), use the following guidelines:

Physical File Name: The CICS file name, up to 7 characters, used to store messages for this queue. A physical file can hold as many queues as desired. A message queue can be logically replenished, if its associated physical file name is changed.

Note: This feature may be used as a emergency short cut in a test environment. For example, a queue file name AAAA, residing in a physical file named P1, gets full. Without deleting and redefining the P1 or using another queue file, the user may simply update P1 to any existing physical file, say P2, and the queue file AAAA will appear as a new file so long as there are no AAAA records hidden in P1.

Maximum Q Depth: The maximum number of records (integer) that can be left unread on this queue. Any inbound message that causes the queue depth to exceed this size will be rejected as “Queue Full”. The value should be set to the maximum number of messages expected to be queued before the application starts to read and process the queue. In practice, it is acceptable to set this to a very large number such as 9999999.

Maximum Message Length: The maximum number (integer) of characters per message that is allowed for this queue. If this queue is a transmission queue, then it needs to be large enough to accommodate all messages using this queue as the outbound queue.

Maximum Concurrent Accesses:

The maximum number (integer) of MQOPENs that can occur to this queue. An unrealistic value can consume too much overhead. A value of 100 would provide a good cushion for any non-transmission queue while consuming little overhead. The

	cushion could help reduce the impact of application program errors that leave opened queues. For a transmission queue, a value of 100 calls should be added to the base of 100 calls for each additional target queue that receives messages from this transmission queue.
Global Lock Entries:	The maximum number of entries (integer) that the Queue Manager uses to maintain committed MQPUT/MQGET activities for this queue for the system for recovery. An integer value equal to or a little less than the Maximum Number of Opens for this queue is what is needed, otherwise valuable resources may be wasted.
Local Lock Entries:	The maximum number of entries (integer) that the Queue Manager uses to maintain uncommitted MQPUT/MQGET activities for this queue for recovery. Since an entry of a Local Lock is deleted once the application issues an explicit SYNCPOINT CICS commands to commit updates, the more often an application takes the checkpoint, the less the Maximum Number of Local Locks is needed. A value of 20 should be sufficient.
Checkpoint Threshold:	The maximum number of queue accesses (integer) between checkpoints to be taken by the Queue Manager for this queue. The lower the Checkpoint Threshold, the more often the Queue Manager takes checkpoints to secure the data integrity and increase the reliability. The greater the Checkpoint Threshold is, the less resources the Queue Manager will consume. A value in the order of 500 to 5000 is a good compromise between performance and reliability.
Trigger Type:	'F' is used to generate a trigger when an MQPUT activity changes the status of a queue from empty to non-empty. The triggered transaction must have 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 activity occurs and may have many threads as specified in Max Trigger Starts.
Maximum Trigger Starts:	The maximum number (integer) of trigger threads that can be active at once. This field is for Trigger Type 'E' only, because Type 'F' supports single threaded processing only.
Trans ID:	The transaction to be started by the trigger. This is mutually exclusive with the Program Id. Leaving this as blank and using a program ID such as MQPSEND is recommended unless a user transaction is desired.
Program ID:	MQPSEND should be used on a transmission queue if triggering is desired.
Term ID:	This field should be left blank unless a terminal is to be used for debugging.

Number of channels per queue manager:

The limit on the number of channels depends more on general resources than on the Queue Manager. The purpose of channel is to access a remote queue. The definition of a remote queue demands the name of transmission queue which in turn is associated with a channel. In other words, as far as Queue Manager is concerned, provided there is enough resource for more transmission queues and channels to be defined, then there can be additional channels.

Example configuration:

Queue manager configuration:

Parameter	Value	Units
Checkpoint Global Timer	60	seconds
Maximum Number of MQCONN	200	integer
Maximum Open Queue	200	integer
System Wait Interval	1	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

Table 19. Example of a Queue Manager Configuration

Channel configuration:

Parameter	Value	Units
Allocation Retries	10	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

Table 20. Example of a Channel Configuration

Queue configuration:

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	MQPSEND	character

Table 21. Example of a Queue Configuration

Chapter 5. Configuring network resources

Introduction

One of the main aims of MQSeries is to allow applications to communicate without knowledge of the lower levels of the communications network. However, this does not mean that, in order to configure the product successfully, no knowledge of the communications network is required. What it does mean, is that this knowledge is not required by the programmer writing the application; underlying network issues can be left to those people in the enterprise who are the experts on the subject.

This chapter is intended to give guidance and general help in configuring the network resources to enable MQSeries to function.

Because of the wide variety of interconnections which may exist (different hardware platforms and a variety of SNA software from a multitude of different third-party vendors), it is not possible here to give comprehensive coverage of every imaginable scenario which might be encountered. However, it is understood that, in any organization installing an MQSeries network, those persons accountable for managing this work may not be familiar with SNA in a mainframe environment. Therefore, the material presented here is intended to give information to clarify what they require when dealing with the organization's technical experts, and to enable them to focus their own further reading more precisely to their present needs.

Authoritative information concerning implementations of the SNA architecture on IBM mainframe systems may be found in the following manuals:

- *VTAM Network Implementation Guide* (SC31-6434)
- *VTAM Resource Definition Reference* (SC31-6438)
- *VTAM Operation* (SC31-6435)
- *VTAM Messages and Codes* (SC31-6433)
- *CICS Resource Definition (Online)* (SC33-0708)
- *CICS Resource Definition (Macro)* (SC33-0709)
- *CICS Intercommunication Guide* (SC33-0701)

Similarly it will be necessary to consult the manuals for the SNA software at the remote system to which the VSE system is being connected. Experience has shown third party SNA software suppliers sometimes have idiosyncratic understandings of certain parts of the SNA architecture, particularly when LU6.2 is involved, and their software reflects this. Therefore, it cannot be taken for granted that what is written in official SNA documentation will hold good in every software implementation. The only solution is to read the manual for the product concerned and gain an understanding of its use.

Further, for those people who possess skills in network configuration, this chapter will provide skeleton resource definitions to illustrate the parameters which need to correspond in the various definitions if the product is to work correctly.

Note: In the skeleton definition outlines provided in this chapter, text in angle brackets, <>, contains a brief description of what is entered in the field. Where the same text is given in angle brackets in different definitions, it means the same value must be coded in both places. Where it is not essential for the same value to be coded, but it is helpful in order to avoid confusion, or it is conventional to do so, a note will explain that <text 1> can equal <text 2>.

Background information

For our purposes here, it is sufficient to note that VTAM (“Virtual Telecommunications Access Method”) provides a software implementation of the entity defined in SNA as a “Systems Services Control Point” (SSCP). In essence, in traditional SNA networks, VTAM is the nerve-centre, monitoring and controlling the network resources.

Where a network consists of several mainframe hosts (“PU Type 5") connected together, each host would have its own copy of VTAM controlling its own resources. This introduced the idea of a VTAM “domain” - that is, all the resources owned (controlled) by a single VTAM. If a resource owned by one VTAM required to communicate with a resource owned by another, the communication is said to be “cross-domain”; thus, in this sort of environment, each interconnected VTAM is seen as providing services as a “cross-domain resource manager” (CDRM), and in any one VTAM domain, resources owned by another VTAM are cross-domain resources (CDRSCs).

An extension to this idea of cross-domain resources comes when different SNA networks need to be connected together (for example, two separate enterprises, or independent divisions in a single enterprise, wish to be able to communicate) thus forming a larger network. Each of the networks being joined is now given a distinct network ID, so the larger network formed is viewed as consisting of several SNA subnetworks, the boundaries being determined by the points where the network ID changes. This scheme enables each subnetwork to manage its affairs relatively independently of any other subnetwork and yet still be able to communicate (for example, across subnetworks, an alias name may be used to refer to a cross-network resource where a name conflict arises). Cross-network resources may be viewed as a special case of cross-domain resources.

Central to SNA is the concept of a session - a logical connection between network components of the different types defined in SNA. Sessions may thus be described as, for example, SSCP-SSCP, SSCP-PU or SSCP-LU, but ultimately, the whole of SNA is concerned with establishing one type of session: those between LUs (that is, LU-LU sessions). In SNA, the term Logical Unit (LU) describes the entry point, or port, for an end-user - an application, or a person at a terminal - to gain access to the network. LU-LU sessions thus represent end-user to end-user connections.

In traditional SNA, each LU was owned by one of the VTAMs in the network: the VTAM activated the LU, establishing an SSCP-LU session between itself and the LU. It was on this session that the LU could send requests to VTAM to initiate a session between the LU and another LU. This type of LU is described as “dependent” since it relies on the services of an SSCP (in this case VTAM) to establish a session to another LU (and, thereby, to another network “end-user”).

With the advent of the requirement for smaller computer systems to be incorporated into networks, it has not been practicable for those systems to support the full functionality of an SSCP, so the ideas of a “Node type 2.1" (to distinquish them from the PU Type 2 nodes which had existed before then) Peripheral Node Control Point (PNCP), and more recently, simply Control Point (CP) were devised (along with the definition of a variety of different types of SNA nodes), and new types of session (for example, CP-CP) were introduced. With these type 2.1 nodes came “independent” LUs - ones which do not need the services of an SSCP (that is, VTAM) to engage in sessions with other LUs.

Independent LUs are not activated by VTAM (there is no SSCP-LU session as there is for dependent LUs), and, even if the node in which they reside (for example, a PC, or a UNIX system) is defined to VTAM as being in its domain, VTAM views the independent LUs as cross-domain resources. Type 2.1 nodes can support both dependent and independent LUs at the same time and it is important to be clear about which sort is involved in any particular case. PU Type 2 nodes (or systems emulating this level of function) can only support dependent LUs.

LU6.2 was introduced to deal specifically with program to program communications (that is, the network end-users are programs rather than people sitting at terminals, although the terminal could be running a program which is engaging in LU6.2 activity). An important concept in LU6.2

communications is that of the “conversation” which is distinct from the SNA “session”, but requires a session in order to take place: a session can exist without a conversation, but a conversation cannot exist without a session; a session is the connection between the LUs, a conversation is the interaction between the applications. This is illustrated in figure 8, “SNA session and conversation” . In theory, a session is a long-lasting resource in that, serially, it can handle many different conversations, one after another. By comparison, a conversation is a short-term resource (although in practice, it may last for a considerable length of time). Note that some LU6.2 implementations, however, terminate the session when the conversation terminates.

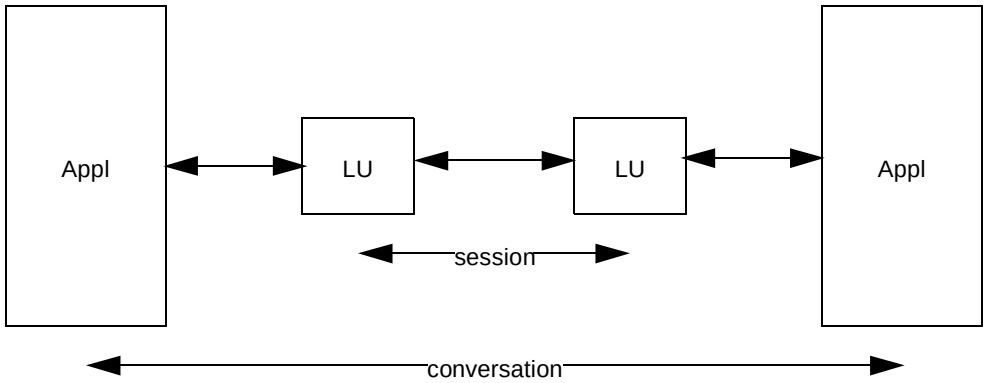


Figure 8. SNA session and conversation

Relating this to MQSeries, it follows that each MQSeries channel requires a single session while the conversation is in progress. Independent LUs are capable of supporting multiple simultaneous sessions between themselves and other LUs. Dependent LUs generally support just one LU-LU session; the CICS LU, however, although dependent, is capable of supporting multiple sessions.

VTAM start up parameter list.

It is highly unlikely that VTAM start parameters will need to be changed for MQSeries System operation. This list is provided to give people with no VTAM experience a better idea of what to ask for when requesting information from those responsible for the SNA software environment.

The startup list(s) are filed with names of the form ATCSTRxx. They will contain a list of entries similar to the outline below:

CDRSCTI = <time out>,	+
HOSTPU = <host pu name>,	+
NETID = <network id>,	+
SSCPNAME = <sscp name>	

Table 22. Extract of ATCSTRxx VTAM start parameters

Notes for table 22:

CDRSCTI is a time-out interval which applies to LUs outside the control of this VTAM (they are “cross-domain”) and which have not been predefined to VTAM as “cross-domain resources” (CDRSCs). Prior to VTAM becoming aware of the existence of these resources (by a session setup request from the VTAM which owns the resource, or from the resource itself if it is an independent LU), the operator command:

```
DISPLAY NET,ID=<lu name>
```

would respond with

```
PARAMETER VALUE INVALID.
```

When a session request is received from the resource or its VTAM, the local VTAM will create a dynamic definition of the resource; the operator command

```
DISPLAY NET,ID=<lu name>,E
```

will now display information VTAM has about this LU (and the sessions it has). When the last session between the dynamically-defined LU and any local LU ends, VTAM retains the resource definition for the length of time specified by CDRSCTI before deleting it, when

```
D NET,ID=<lu name>
```

will again respond with

```
PARAMETER VALUE INVALID.
```

The command

```
D NET,ID=ISTCDRDY
```

will provide details of all the resources dynamically defined by VTAM (but be warned it could be a very long list!). A corollary of this is that, if the session setup request is issued by CICS when a dynamic definition is NOT in place, VTAM will not know the resource, and, although it has means to attempt to find it, these may fail resulting in a session establishment failure (normally with a 087D sense code). This has implications for MQSeries if the CICS end of a channel is to start the channel and the session has not already been established: if the session cannot be started, the MQSeries System software cannot conduct a conversation on it!

HOSTPU is the SNA Physical Unit name of the Physical Unit where CICS resides.

NETID is the network ID in which this VTAM and all the resources it controls reside (that is, the network ID of this VTAM domain). Note the following points:

There is a distinction between the similar sounding terms “net-ID” and “netname”. “net-ID” specifically refers to the name of the network in which a resource resides; “netname” refers to the name of the resource itself, as it is known in the network - it could be an luname, a puname, a cdrscname, an sscpname, etc., depending on the type of resource concerned.

A “fully qualified network name” (frequently required by SNA software on non-mainframe systems) consists of both the netid and the netname, separated by a period. Thus, the fully qualified network name of the CICS system (see table 22) is <network id>.<minor node name> .

SSCPNAME is the name of this VTAM. It may be used in a definition at a remote VTAM to identify this VTAM as an adjacent SSCP (see table 22).

Definition of CICS to VTAM

It is unlikely that the MQSeries System will require any changes to the definition of the CICS system to VTAM. However, it is worthwhile reviewing the existing definition to ensure nothing is amiss. CICS is defined to VTAM by an entry in an “application major node”. It will look similar to the outline in table 23.

```
<major node name> VBUILD TYPE = APPL
*
<minor node name> APPL ACBNAME = <acb name>, APPC=YES                +
                       AUTH = (ACQ,PASS,...),                        +
                       EAS = ?,                                      +
                       PARSESS = YES,                                +
                       SONSCIP = YES,                                +
                       MODETAB = <cics mode tab>,                    +
                       MODENT=
```

Table 23. Skeleton VTAM definition for CICS

Notes for table 23:

APPC=NO is coded for CICS, or, as above, allowed to default (this is usual for CICS, and does not depend on the presence or otherwise of MQSeries System software).

<minor node name>

is the “CICS LU name”, that is, the LU name a remote MQSeries System should target if it wishes to start a channel to the MQSeries System running on this CICS. The distinction between <minor node name> and <acb name> is that <minor node name> can be accessed from anywhere in the entire interconnected SNA network(s); <acb name> is only known in the domain of the VTAM where CICS runs. Normally, to avoid confusion, <minor node name> and <acb name> are the same value (or ACBNAME= is not coded).

EAS is an estimated number of simultaneous sessions between CICS and other LUs of all types. If a large number of the MQSeries System channels is to be defined, the value coded here may need to be reviewed.

PARSESS=YES indicates the CICS LU is capable of parallel sessions (multiple sessions to the same partner LU).

APPL Further APPL definition statements may exist under the same major node name, for other CICS systems at this host, or for other VTAM applications in general at this host.

For CICS to use this definition, its own startup parameters have to be set up to cause it to OPEN the correct ACB. This is done by coding an entry in the CICS System Initialization Table (SIT), or by providing a SIT override at CICS startup. See table 24.

```
APPLID = <acb name>
```

Table 24. CICS SIT parameter

The APPL definition in table 23 made reference to a “mode table” (MODETAB). A logon mode table specifies details of the LU. For VTAM, these tables have to be assembled from a source deck and then brought into memory. The source, which may exist anywhere on the host system, will look similar to that in table 25 below.

```
<cics mode tab> MODETAB
*
                MODEENT LOGMODE = <cics mode 1>
*
                MODEENT LOGMODE = <cics mode 2>
                .
                .
                .
                MODEEND
```

Table 25. Skeleton logon mode table source

Notes for table 25:

A mode table may contain definitions of several different logon modes (two separate modes, called <cics mode 1> and <cics mode 2> are shown in the table above).

Definitions required for the remote MQSeries System

The remote MQSeries System has to be defined:

- to MQSeries on CICS (in the network specific parts of the channel definition)
- to CICS itself (in a TERMINAL definition, or in CONNECTION/SESSION definition(s), or by the CICS AUTOINSTALL facility)
- to VTAM (either predefined, or by VTAM dynamic resource definition).

MQSeries System channel definition

Defining the remote MQSeries System to the local Queue Manager is covered in this manual in Chapter 5, “System Operation”, and will not be discussed further here. However, from the point of view of showing where fields in the various definitions have to correspond, a skeleton MQSeries System channel definition is shown in figure 9 below.:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:51:28                    Channel Record           DISPLAY         MCHN
MQMMCHN     Last Check Point               Last Update  19961211    0002
MSN 00000009  Time 10:28:31  Interv 000000  Create Date  19961128

 Channel Name  : VSE1_TO_VSE2               Channel Type  : R Snd,Srv/Rcv
 Protocol  : L62    L62                     Format : MCP    MLP/MEP/MCP

        Allocation Retries               Get Retries
    Number of Retries: 00000005       Number of Retries : 00000001
    Delay Time - fast: 00000015       Delay Time        : 00000001
    Delay Time - slow: 00000003

  Max Messages per Batch : 000001         Max Transmission Size   : 004096
  Message Sequence Wrap  : 999999         Max Message Size        : 003338

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

Figure 9. Skeleton MQSeries channel definition

Notes for figure 9:
Remote task ID need only be coded for a SENDER channel.

Definitions in CICS

If the CICS end of an MQSeries System channel is to initiate the channel connection (that is, the CICS channel-endpoint is a SENDER), then CICS will perform an EXEC CICS ALLOCATE. Note, however, this will only succeed if it is:

- a contention winner
- already bound
- not already allocated.

If CICS has no definition of the resource, it is incapable of formulating a request to VTAM for session establishment. This means, in these circumstances, CICS AUTOINSTALL is inappropriate (autoinstall is for incoming session establishment requests, not for outgoing ones).

Thus, 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 it is the intention to configure several channels between the two systems, the most natural definition for this system in CICS is to have CONNECTION and SESSIONS definitions. Typical definitions, using the CICS Resource Definition Online (RDO) transaction, CEDA, is shown in table 26.

```
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>)}
```

Table 26. 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 (table 27) or as a terminal definition (table 28).

```
 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>)}
```

Table 27. 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>)}
```

Table 28. Definitions in CICS singles-session capable LU

Notes for table 28:
The CICS supplied typeterm definition, DFHLU62T, provides a suitable terminal type definition. It exists in group DFHTYPE, which should be installed on your system.

Definitions in VTAM or NCP

Just as CICS is not able to pass a session establishment request to VTAM if it has no definition of the remote LU which is to be the target of this request, if VTAM is not able to identify the remote LU, the session establishment will fail, and, consequently, the attempt to start the channel will fail.

There are several strategies to avoid session establishment failure:

- Ensure sessions are established prior to an attempt to start the channel. Sessions can be started when the SNA support software at the interconnected systems is started, even though a conversation will not immediately be carried out on the sessions. It is possible for a remote system to bind sessions which will subsequently be used by CICS to initiate a channel connection. Note, however, some SNA implementations may impose limits on this - it may not be possible at all to establish a session without also issuing an LU6.2 ALLOCATE command, or a remote system may only be capable of binding those sessions for which it is the “contention winner”.
- Ensure VTAM has sufficient information to be able to locate the remote LU. This can involve use of one or more of the following types of resource definitions:
  - ADJACENT SSCP TABLES: details of other VTAMs to which the local VTAM can make enquiries to see if they know the target LU.
  - CROSS-DOMAIN RESOURCE DEFINITION: a System Programmer definition of a resource which is not activated by the VTAM at this host; that is, this host's VTAM does not have an SSCP-LU session with the resource. The resource could be an independent LU6.2 in this or another domain, or a dependent LU6.2 in another domain (a dependent LU6.2 in this domain would be explicitly defined elsewhere).
  - APPLICATION DEFINITION: if the remote queue manager is another CICS system residing on the same host, this is the only definition which will be necessary to completely define the system to VTAM. The definition will be similar to that already shown for this CICS.
  - CHANNEL ATTACHMENT MAJOR NODE: for other host systems which are channel attached to this host.
  - LOCAL MAJOR NODE: For peripheral nodes which are channel attached to the host.
  - NETWORK CONTROL PROGRAM (NCP) MAJOR NODE: For peripheral nodes accessed via a communications controller.

The details of the majority of these definitions are beyond the scope of this User's Guide; the network Systems Programmer will know what is appropriate for the particular circumstances. However, since, in a large number of cases, the remote Queue Manager's LU will reside in a peripheral node of one sort or another, details of how this might be defined will be covered.

If the remote LU is an independent LU, it may either be defined to the local VTAM as a cross domain resource, or it may be defined, along with any dependent LUs for the node, in either a Local major node, or an NCP major node, depending on whether the peripheral node is channel attached to the host, or attached via a communications controller. Whichever method is used, VTAM collects together all the information it has about Pre-Defined Independent LUs into one major node, ISTPDILU, where each independent LU is a separate minor node. Thus the VTAM command:

```
D NET,ID=ISTPDILU,E
```

will list all the predefined independent LUs known to this VTAM. (Dynamically defined independent LUs will be listed under ISTCDRDY, along with other dynamically defined resources, and will remain known for the period given by CDRSCTI, as noted above.) There may be a large number of resources listed, so use the command with caution.
If the remote LU is a dependent LU, it should only be defined as a cross-domain (or cross-network) resource if it is not in the domain of VSE's VTAM. Otherwise, it will be defined in a local major node, or NCP major node as appropriate, as in table 29 below.

<remote luname> LU LOCADDR = <address>, MODETAB = <rem mode tab>, DLOGMOD = <logmode 1> or <logmode 2>	+ +

Table 29. Local or NCP Major Node definition of the remote LU

Notes for table 29:

<remote luname> matches the value given in the definition in CICS. This is how VTAM relates the request from CICS to the resource intended.

<address> should be coded as 0 for an independent LU, and nonzero in accordance with normal VTAM/NCP conventions, for a dependent LU. Very often, the SNA software on PU Type 2 or Type 2.1 nodes will use this value, rather than the resource name, <remote luname> (which will be considered the "local" luname from their point of view), to relate an incoming request to a particular LU. Such software often refers to this value as the "LU number" or something similar. The number coded at the remote SNA software must be <address>. With this software, it is not necessary for <remote luname> to match the name used in the remote SNA software as "local LU name", but it will avoid confusion if it does.

Table 30 shows a typical Logon Mode Table which might be coded for the LU shown in Table 29, “Local or NCP Major Node definition of the remote LU,” on page 60. A complete definition from a working channel is also shown as the entry <practical example>..

```
 <rem mode tab>  MODETAB
*
                 MODEENT LOGMODE=<logmode 1>,                          +
                 ...
*
                 MODEENT LOGMODE=<logmode 2>,                          +
                 ...
*
                 MODEENT LOGMODE=<practical example>,                  +
                       FMPROF=X'13',                                   +
                       TSPROF=X'07',                                   +
                       PRIPROT=X'B0',                                  +
                       SECPROT=X'B0',                                  +
                       COMPROT=X'50B1',                                +
                       TYPE=0,                                         +
                       PSNDPAC=X'00',                                  +
                       SRCVPAC=X'00',                                  +
                       SSNDPAC=X'00',                                  +
                       RUSIZES=X'8888',                                +
                       PSERVIC=X'060200000000000000002F00'
*
                 MODEEND
                 END
```

Table 30. Skeleton Logon Mode Table for the remote LU

Definitions on the remote SNA software

It is possible to deal only in very broad terms of the definitions which may be required on the remote system's SNA software.

If the remote system is an IBM mainframe, the types of definitions will be very similar to those explained for CICS on the VSE host.

As a very general guide, for other systems (for example, UNIX based) table 31, below, gives a list of the resources which would need to be defined on these. Under the heading “Value to code” in table 31, the term “remote” has been used with the same meaning as established earlier - as being remote from the CICS host's point of view. However, from the point of view of the system on which the definitions are entered, it is “local” and CICS is “remote”.

Likely resource type on remote SNA software	Value to code
Fully Qualified Local LU name	<remote net-ID>.<remote lu name>
Local LU alias name	<remote lu name>
LU number (for dependent LU6.2)	<address>
Fully qualified partner LU name	<net-ID>.<minor node name>
Partner LU name	<minor node name>
Mode name	<logmode 1> or <logmode 2>
Max number of sessions	<max sessions>
Maximum local contention winners	<max sessions> minus <max cics contention winners>
Maximum remote contention winners	<max cics contention winners>
Transaction Program	<remote TP>

Table 31. Values to code in the remote SNA software

Notes:

<remote net-ID> The network ID of the SNA subnetwork in which this system resides. If the LU is to be treated as being in the same SNA subnetwork as CICS, this should be the same as <net-ID>. If the LU is a dependent one requiring CICS' VTAM to activate it in order to establish sessions, it must be in the same subnetwork as CICS.

<remote TP> unlike on CICS, on many Type 2.1 nodes a UNIQUE TP name must be defined for each active channel in order to allow the LU to direct network traffic to the correct MCA for the channel. TP names are required only for channels defined as RECEIVER (or SERVER) types at the Type 2.1 node. (Since REQUESTER channels are not supported on VSE, there will not be a SERVER definition at this node.)

Troubleshooting

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 prior to starting the channel, or as a concomitant of it, the channel will not start.

If a session failure is suspected to be at the root of a channel startup failure, enter the following VTAM command:

```
D NET,ID=<remote lu name>,E
```

This will give details of the LU which should be in session with CICS, and will also list any sessions it currently has. Note the session limit for the LU: if it is shown as one for an independent LU, there is a problem with the SNA definitions. See if <minor node name> is listed amongst the sessions. If it is, there is a session between the LU and CICS, indicating that the problem may not be at the network level, or if it is, it might be that there are not enough sessions between the two LUs to support a new channel request. Enter the command again, and see if, 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 VTAM does not know of <remote lu name>: it was either entered incorrectly, or it cannot be located. If the latter, define it to VTAM and attempt to start the channel again.

If VTAM was able to display <remote lu name>, try the following commands in CICS:

```
CEMT I CONN(<remote conn>)
```

This shows the status of the connection from CICS to the remote system. Beside the entry will be an indication showing it to be INService or OUTservice and ACQuired or RELeased. It needs to be Inservice and Acquired.

```
CEMT I MODE CONN(<remote conn>)
```

This 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 will be RELeased, rather than ACQuired. These sessions are SNA services manager sessions, not used by MQSeries System channels, but 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 no other sessions can be established, including those required by the MQSeries System channel. The <logmode 1> sessions may be used by MQSeries System channels.

For single session connections, one mode name, <logmode 2>, will be shown with just one session in the group.

The MQSeries System channel must have been set up to use the logon mode <logmode 1>, or <logmode 2>, as appropriate.

Chapter 6. System operation

This chapter will describe the system operation and administration functions available in IBM MQSeries for VSE/ESA. Most such functions are provided through the menu driven, screen oriented program associated with the CICS transaction **MQMT**. An additional CICS transaction is a command line module that performs actions against the runtime queues and channels. These actions include stopping and starting queues and closing and opening channels.

Background functions include the queue dump and system utility facilities. The queue dump facility allows a user to rebuild a MQSeries System VSAM queue file. This eliminates processed messages and fully regains VSAM freespace. The utility function includes the ability to print MQSeries System Configuration, System Log messages and the Help facility information. This utility function also includes the ability to reset the same Message Sequence Number to all of the channel definitions and change all Dual queue definitions to primary queues.

The menu's and display screens of *MQMT* 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/return paths available from lower level screens.

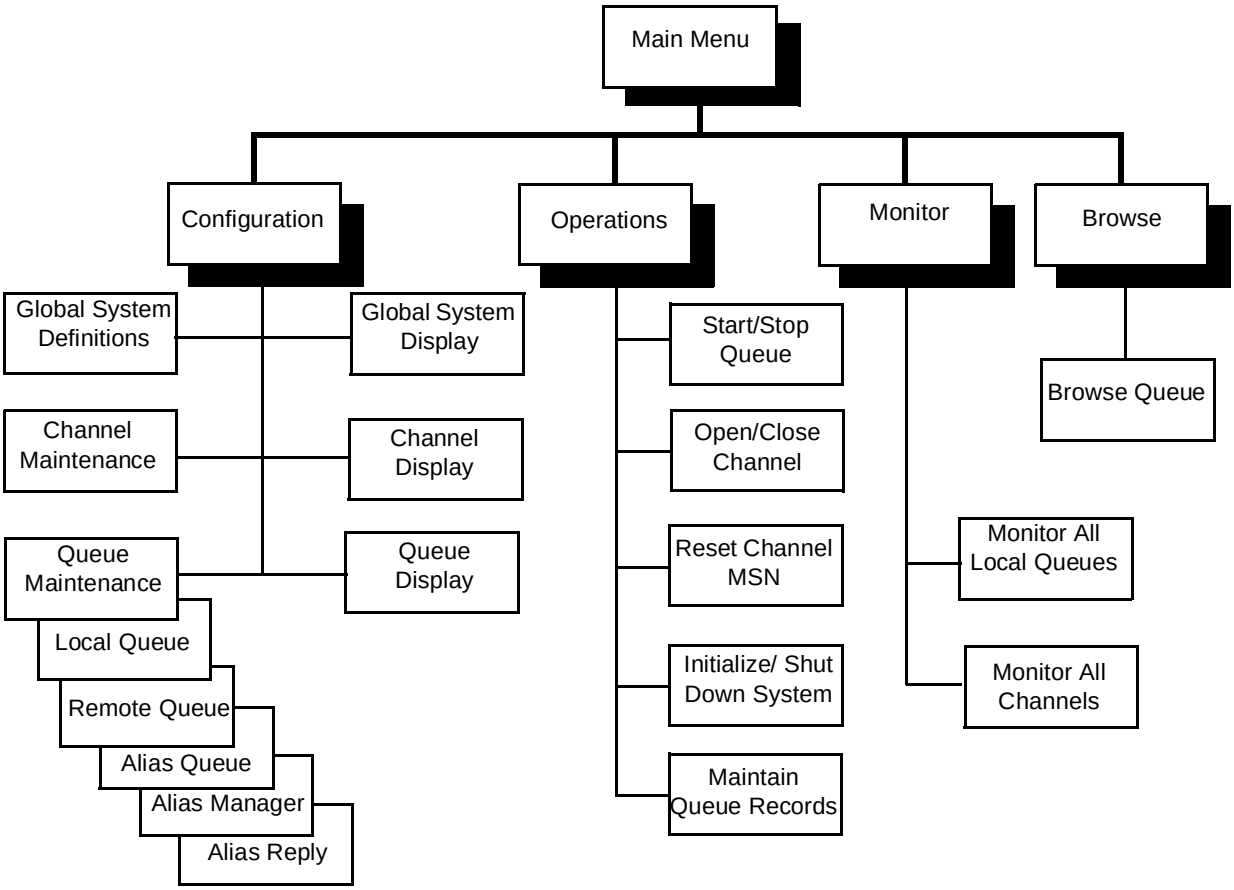


Figure 10. Display screen relationships

In the next section, the main *MQMT* menu is presented. The subsequent sections will present each of the operator functions available through these screens. The final section in this chapter will present those functions which require operator action outside *MQMT*.

General panel layout

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

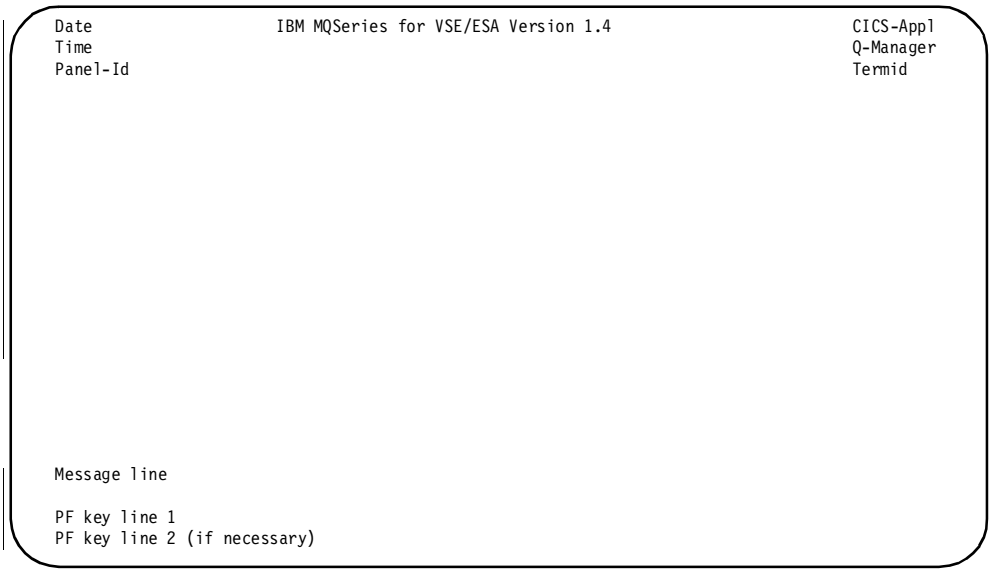


Figure 11. General panel layout

Where:

CICS-Appl: The VTAM application Id for this CICS partition

Panel-Id: The panel name which is displayed.

Q-Manager: The name of the MQSeries Queue Manager specified in the Global Definitions.

Termid: The CICS terminal Id on which this panel is displayed.

MQMT master terminal - main menu

The MQSeries System administrator program, *MQMT*, may be invoked from any 3270 terminal. To access the operator functions, simply type the following at the CICS prompt:

```
MQMT
```

When MQMT starts, the main menu is displayed.

```
01/28/1997        IBM MQSeries for VSE/ESA Version 1.4              IYZMZSI2
08:36:59           *** Master Terminal Main Menu ***                VSE2
MQMMTP                                                              0002
                     SYSTEM HAS BEEN SHUTDOWN

                           1. Configuration

                           2. Operations

                           3. Monitoring

                           4. Browse Queue Records


                    Option:




Please enter one of the options listed.
    5787-ECX (C) Copyright IBM Corp. 1993, 1997  All Rights Reserved.
    CLEAR/PF3 = Exit                          ENTER=Select
```

Figure 12. Master terminal main menu

From the Main Menu, one of several sub-menus may be selected. The first three choices correspond to broad categories which include most MQSeries System operator functions:

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

The fourth function allows the operator to display the records on a selected queue.

- Browsing MQSeries System Queues

Each sub-menu 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 System 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

PF2 = Return to Prior 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 = Varies by Screen

PF12 = Delete

Configuration functions

Selecting option 1 (Configuration) from the main menu, causes MQMT to display the following sub-menu screen:

```
01/28/1997        IBM MQSeries for VSE/ESA Version 1.4              IYZMZSI2
08:39:08           *** Configuration Main Menu ***                  VSE2
MQMMCFG                                                             0002
                            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.
     5787-ECX (C) Copyright IBM Corp. 1993, 1997  All Rights Reserved.
  ENTER = Process          PF2 = Main Menu             PF3 = Quit
```

Figure 13. Configuration main menu

On this screen, choices 1 through 3 allow the operator to perform maintenance functions on various MQSeries System configuration objects. Choices 4, 5, and 6 allow the passive viewing of the same objects.

Notes:

1. Changes to parameters on Configuration screens only take effect when the queuing system is re-initialized.
2. When values are shown on the screens, they are default values.

Global system definition

For each installation of the MQSeries System, one and only one Queue Manager must be defined. This is accomplished through the screen below. This screen is also used to modify previously defined global parameters.

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:52:14                  Global System Definition                  VSE2
MQMMSYS                  Queue Manager Information                  0002
 Queue Manager . . . . . .: VSE2
 Description Line 1. . . . .: MQ/Series Manager on VSE/ESA 2.1
 Description Line 2. . . . .: Development System
                       Channel Maximum Values
 Checkpointer Global Timer .: 00000060
                       Queue System Values
 Maximum Number of Tasks . .: 00000500      System Wait Interval : 00000030
 Maximum Concurrent Queues .: 00000500      Max. Recovery Tasks  : 0001
 Allow TDQ Write on Errors  : Y   CSMT      Allow Internal Dump  : N
                       Queue Maximum Values
 Maximum Q Depth . . . . . .: 01000000      Maximum Global Locks.: 00000100
 Maximum Message Size. . . .: 00032000      Maximum Local Locks .: 00000100
 Maximum Single Q Access . .: 00000100      Checkpoint Threshold : 1000
                       Global QUEUE /File Names
 Configuration File. . . . .: MQFCNFG
 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
```

Figure 14. 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 MQI naming requirements.

Description Lines 1 & 2: This is a text field for operator use only. It may be up to 64 characters.

Channel Maximum Values

Checkpointer Global Timer: The time interval set to take a global checkpoint of all active channel parameters.

Queue System Values

Maximum Number of Tasks: The maximum number of simultaneous connections to the queue manager.

Maximum Concurrent Queues: The maximum number of simultaneous open queues.

Allow TDQ Write on Errors: Y - allow writes to the CICS TDQ 'CSMT' if SYSTEM.LOG not available
N - do not allow TDQ write.
B - write to both SYSTEM.LOG and the 'CMST' TDQ.

Max. Recovery Tasks: Maximum number of tasks attached by the System Monitor when errors are detected in queues or control blocks attached to queues. An high number would lead to use too many CICS resources and have a negative impact on the overall CICS performance. The suggested value is 1.

System Wait Interval: The sleep time in seconds for the system monitor program and startup of trigger programs after system initialization.

Allow Internal Dump: Allow the API to execute a CICS Task Dump if the internal area(s) is(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 Hobj allowed for a queue.

Maximum Global Locks: The maximum number of entries that the queue manager uses to maintain destructive PUT/GET locks, per queue, for the system.

Maximum Local Locks: The maximum number of entries that the queue manager uses to maintain destructive PUT/GET locks, per queue, for each individual task.

Checkpoint Threshold: The maximum number of queue accesses between checkpoints.

Global QUEUE/File Names

Configuration File: The CICS file definition name of the MQSeries System configuration file.

LOG Queue name: The queue name where the MQSeries System 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. See "CMQDLHV.C" on page 269.

Monitor Queue name: The queue that the API application requests when the System Monitor is turned on.

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.

Queue definitions

Choice 2 on the configuration menu allows an operator to maintain (add, modify, or delete) queue definitions for the local installation of the MQSeries System.

Note: The same screens are used to accomplish all three functions (add, modify, or delete), with the desired action being indicated via the function keys. The following sections will present screens as if the operation is to **add** a new queue definition. “Modifying and deleting queue definitions” on page 78, presents the slightly different operation to modify or delete a queue.

To create a queue definition, multiple screens may be involved. The first screen is the same for all queues. It allows entry of the queue *name* and *type*. Based on the *type* entered, the appropriate second screen is displayed for the operator to enter the remainder of the data to complete the definition. In the case of Local queues, a third screen will also be involved. This third screen is the Extended Queue Definition Screen. The first screen displayed is:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4             IYZMZSI2
11:39:27                Queue Main Options                           VSE2
MQMMQUE                                                              0002
                              SYSTEM IS ACTIVE

        Default Q Manager  : VSE2

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

                Object Name: QUE.TEST









Function has been terminated.

    PF2=Main Config   PF3 = Quit PF4/ENTER = Rea PF5 = Add       PF6 = Update
                                                 PF9 = List      PF12= Delete
```

Figure 15. Queue main menu 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 follow on 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 MQI naming requirements.

Upon entry of the above two fields, the *Object Type* is used to determine which of the following five screens is displayed:

Create local queue

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:40:18                Queue Definition  Record                    VSE2
MQMMQUE           QM - VSE2                                         0002

                       LOCAL QUEUE DEFINITION

 Object Name. . . . . . . . : QUE.TEST
 Description line 1 . . . . : Transmission Queue for testing
 Description line 2 . . . . : MQSeries/VSE

 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 . . . . .:



Record being added   - Press ADD key again.

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

Figure 16. Local queue definition

On this screen the data entry fields are:

Object Name: Filled in from the previous screen.

Description Lines 1 & 2: Text field for operator use only. It may be up to two 32 character fields.

Put Enabled: This is a toggle which enables/disables `MQPUT` operations against this queue.

Get Enabled: This is a toggle which enables/disables `MQGET` operations against this queue.

Default Inbound status: This sets the initial status to Active or Inactive at run time for the Inbound direction of the queue.

Outbound status: This 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 will become the primary and the queue entered in this field will become the dual queue. The definition of the dual queue will be updated automatically with the name of the primary queue. The queue display of the dual queue will have a corresponding heading “Dual Source Queue”.

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

Note: Once an existing queue is defined as the dual to a primary queue, these two queues both participate in the same logical unit of work. If for any reason, it becomes impossible to update the dual queue (for example, if the queue becomes disabled, the associated file 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.

By pressing PF10, the operator may bring up a second screen to enter the extended definition fields for the queue. On an ADD request, this Extended Definition Screen will be presented automatically. This detailed screen is:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:41:47                Queue Extended Definition                   VSE2
MQMMQUE           QM - VSE2                                         0002
Object Name. . . . . . . . : QUE.TEST
                          Physical Queue Information
Usage Mode . . . . . . . . : T     N=Normal, T=Transmission
Share Mode . . . . . . . . : Y     Y=Yes, N=No
Physical File Name . . . . : MQFI003    MQSERIES.DEVELOP.MQFI003
                             Maximum Values
Maximum Q Depth. . . . . . : 01000000     Global Lock Entries . : 00000100
Maximum Message Length . . : 00003353     Local Lock Entries. . : 00000100
Maximum Concurrent Accesses: 00000100     Checkpoint Threshold  : 1000

                          Trigger Information
Trigger Enable . . . . . . : Y     Y=yes, N=No
Trigger Type   . . . . . . : E     F=First, E=Every
Maximum Trigger Starts . . : 0001
Allow Restart of Trigger   : N     Y=Yes, N=No
Trans ID   :           Term ID :          SYSID       :
Program ID : MQPSEND                      Channel Name: VSE2_TO_VSE1

Record added OK.
    PF2 = Options   PF3 = Quit     PF4/ENTER= Read    PF5 = Add  PF6 = Update
                    PF9 = List     PF10 = Queue                  PF12 = Deletes
```

Figure 17. 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 means the queue is used by an application to receive inbound messages. Transmission means the queue is used by the MQSeries System to hold outbound messages destined for another MQSeries System queue manager.

Share Mode: Defines a queue as shareable or exclusive on input.

Physical File Name: The CICS file name used to store messages for this queue. 7 characters, max.

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: This is the maximum number of MQOPENs that can occur for this queue at once.

Global Lock Entries: This is used to allocate the locking table for this queue for all committed MQGETs

Local Lock Entries: This is used to allocate the locking table for this queue for each task's non-committed MQGETs

Checkpoint Threshold: The maximum number of queue accesses between checkpoints.

Trigger Information

Trigger Enable: Yes or No. If defining a transmission queue for use with a Sender channel, set this value to Y; for use with a Server or Receiver set this field to N.

Trigger Type: F: 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 following field (Max Trigger Starts). Only one transaction can be active against the queue if the Trigger Type = F.
Maximum Trigger Starts:	The maximum number of trigger threads that can be active at once.
Allow Restart of Trigger:	This field allows the automatic restart of an application if the trigger count goes to zero. It will restart one trigger if messages are available on this queue.
Trans ID:	The name of the transaction to be started by a trigger. 4 characters. If a transaction id is specified, this transaction will be “STARTed” with the communications area passed via “RETRIEVEd”. If defining a transmission queue, this field will be left blank.
Program ID:	The name of the user program to be invoked, 8 characters. If defining a transmission queue to be used with a Sender channel, MQPSEND must be used. If the field for Transaction ID is left blank and this field contains a program ID, then the specified program will be invoked by “LINKed” with data passed via COMMAREA.
Term ID:	Optional. Used for debugging. To be attached to the Transaction ID specified above. 4 characters.
SYSID:	Reserved for future use.
Channel Name:	Identifies the channel name. 20 characters.

Notes:

1. The PF10 key can be used to toggle between the Local Queue Definition screen and the Local Queue Extended Definition screen.
2. One of the items marked with a ‡ is required if the trigger is enabled. If transaction id is specified, this transaction will be “STARTed” with the communication area passed via “RETRIEVEd”, while the program-id is to be “LINKed” with COMMAREA.
3. The internal MQSeries System trigger API transaction MQ02 cannot be used as a Trigger Transaction ID. This is implied when only a trigger program is defined.
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 Comm. area (See “Triggers” on page 109, for more details).
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 IBM MQSeries System header and cannot be bigger than the VSAM CISIZE-7.
6. For information on configuring a transmission queue for a Sender/Server channel, see “Triggers” on page 109.

Navigation through the screens is dependent upon the PF keys.

Create remote queue

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:44:47               Queue Definition  Record                     VSE2
MQMMQUE           QM - VSE2                                         0002

                      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 18. Remote queue definition

On this screen the data entry fields are:

Object Name: Filled in from the previous screen.

Description Lines 1 & 2: Text field for operator use only. It may be up to two 32 character fields.

Put Enabled: This is a toggle which enables/disables `MQPUT` operations against this queue.

Get Enabled: This is a toggle which enables/disables `MQGET` operations against this queue.

Default Inbound status: This sets the initial status to Active or Inactive at run time for the Inbound direction of the queue.

Outbound status: This 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 *Remote Queue Name* is defined as a local queue. This name must be defined as a local transmission queue unless the following field us used.

TRANSMISSION Q NAME: The name of the local transmission queue to be used by the MQSeries System to convey messages to this remote queue. If left blank then the *Remote Queue Manager Name* is required to map to a local transmission queue.

Note: Some other operating systems, which the MQSeries System for VSE user may be communicating with, may be case sensitive. It is important to read “Uppercase translation”, on page 9, before devising a name for a queue, channel or Queue Manager.

Navigation through the screens is dependent upon the PF keys.

Create alias queue

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:47:16                Queue Definition  Record                    VSE2
MQMMQUE           QM - VSE2                                         0002

                      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 19. Alias queue definition

On this screen the data entry fields are:

Object Name: Filled in from the previous screen.

Description Lines 1 & 2: Text field for operator use only. It may be up to two 32 character fields.

Put Enabled: This is a toggle which enables/disables `MQPUT` operations against this queue.

Get Enabled: This is a toggle which enables/disables `MQGET` operations against this queue.

Default Inbound status: This sets the initial status to Active or Inactive at run time for the Inbound direction of the queue.

Outbound status: This 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

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:48:28               Queue Definition  Record                     VSE2
MQMMQUE           QM - VSE2                                         0002

                      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 20. Alias queue manager definition

On this screen the data entry fields are:

Object Name: Filled in from the previous screen.

Description Lines 1 & 2: Text field for operator use only. It may be up to two 32 character fields.

Put Enabled*: This is a toggle which enables/disables `MQPUT` operations against this queue.

Get Enabled*: This is a toggle which enables/disables `MQGET` operations against this queue.

Default Inbound status: This sets the initial status to Active or Inactive at run time for the Inbound direction of the queue.

Outbound status: This 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 transmit 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 the MQSeries System to convey messages to this remote queue manager. If left blank then the above field is required to map to a local transmission queue or to the local queue manager name.

Notes:

1. The above definitions cannot be used in a MQCONN call. They may only be used for `MQOPEN` substitution.
2. The field definitions marked with a * are non-enterable fields.

Navigation through the screens is dependent upon the PF keys.

Create alias reply

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:49:45                Queue Definition  Record                    VSE2
MQMMQUE           QM - VSE2                                         0002

                      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 21. Alias queue reply definition

On this screen the data entry fields are:

Object Name: Filled in from the previous screen.

Description Lines 1 & 2: Text field for operator use only.

Put Enabled*: This is a toggle which enables/disables `MQPUT` operations against this queue.

Get Enabled*: This is a toggle which enables/disables `MQGET` operations against this queue.

Default Inbound status: This sets the initial status to Active or Inactive at run time for the Inbound direction of the queue.

Outbound status: This 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 transmit queue name* or a *remote queue manager name*. It cannot identify another alias.

Notes:

1. The above definitions cannot be used in the `MQOPEN` call. They may only be used for Reply Queue name substitution with a `MQPUT` call.
2. The field definitions marked with a * are non-enterable fields.

Navigation through the screens is dependent upon the PF keys.

Modifying and deleting queue definitions

Choice 2 on the configuration menu (the same option as for creating a queue) also allows an operator to modify, or delete queue definitions.

Note: The same primary screens are used in the modify and delete operations as were described above for the add function. The PF6 key is used to modify existing definitions. These screens are not represented here. However, the “LIST” screen is presented and the “flow” for the modify and delete operation is described.

Selecting an existing queue definition

To modify or delete an existing queue definition, the operator must first select the definition on which to work and bring it to the display screen. This can be accomplished by using either of two function keys.

From the “QUEUE MAIN MENU” screen (this is the first screen displayed after choosing option 2 on the configuration menu), the operator may use either PF4 or PF9.

PF4 is the READ key. It may be used 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. The MQSeries System will read and display the queue definition corresponding to the entered name.

PF9 is the LIST key. It may be used to bring a specific queue definition to the screen as follows:

1. Press PF9.
2. The MQSeries System will display a list of all defined queues (see screen below).
3. The operator selects the desired queue by typing an “X” next to the desired queue or by placing the cursor on the appropriate object.
4. Press PF4 or Enter.
5. The MQSeries System will read and display the queue definition corresponding to the selected entry.

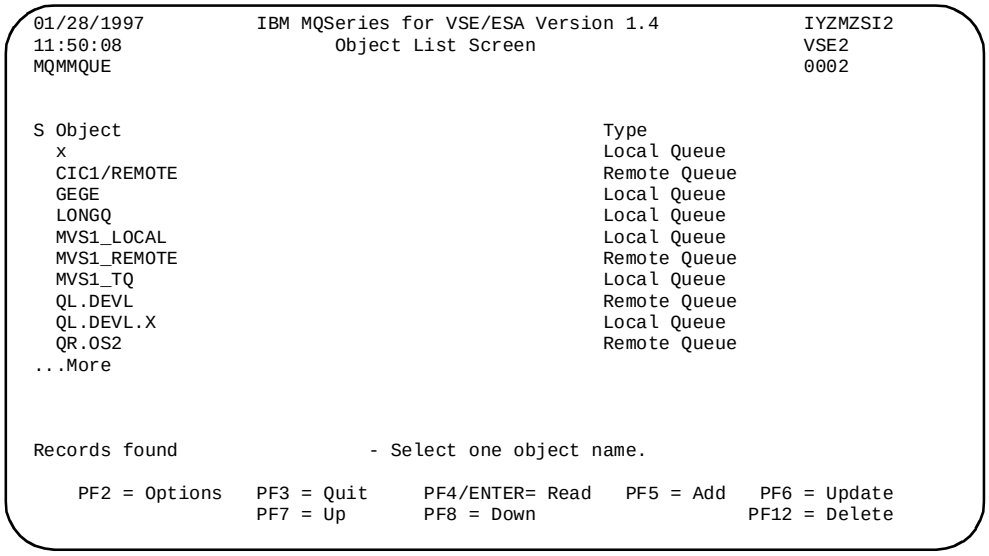


Figure 22. Queue list screen

Modifying an existing queue definition

Once the desired queue definition has been brought to the display (as described in “Selecting an existing queue definition” on page 78), any field of the definition may be modified just as described in the preceding section for the *add* operation. This may involve multiple screens to include all fields of the queue definition

When the desired changes have been made, the operator updates the screen via PF6 (=UPDATE).

Deleting an existing queue definition

Once the desired queue definition has been brought to the display (as described in “Selecting an existing queue definition” on page 78), it may be deleted by pressing PF12 (=DELETE). A confirm request will be presented upon which PF12 must be pressed again.

Channel definitions

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

Note: The same screen is used for all three functions (add, modify, or delete), with the desired action being indicated via the function keys. The following section will present screen as if the operation is to **add** a new channel definition. “Modifying and deleting channel definitions” on page 82, presents the slightly different operation to modify or delete a channel.

To create a channel definition (in response to choice 3 on the configuration menu), the following screen is displayed:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4             IYZMZSI2
11:51:28                    Channel Record           DISPLAY         MCHN
MQMMCHN     Last Check Point               Last Update  19961211    0002
MSN 00000009  Time 10:28:31  Interv 000000  Create Date  19961128

 Channel Name  : VSE1_TO_VSE2                Channel Type  : R Snd,Srv/Rcv
 Protocol  : L62    L62                      Format : MCP    MLP/MEP/MCP

       Allocation Retries                Get Retries
   Number of Retries: 00000005        Number of Retries : 00000001
   Delay Time - fast: 00000015        Delay Time        : 00000001
   Delay Time - slow: 00000003

  Max Messages per Batch : 000001         Max Transmission Size   : 004096
  Message Sequence Wrap  : 999999         Max Message Size        : 003338

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

Figure 23. Channel record

On this screen the data entry fields are:

Channel Name: The name of the channel to be defined.

Protocol: The protocol being used by the selected channel (only L6.2 is supported).

Channel Type: S: A sender/server only channel. R: A receiver only channel. Requester Channels are not supported for IBM MQSeries for VSE/ESA.

Format: Identifies the channel format (only MCP is supported).

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 have 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 (only one message per batch is supported).

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 cannot be bigger than the application message size plus the IBM MQSeries System header.

Mess Seq Reqd: If yes, both ends of the channel must use message sequence numbers. If no, message sequence numbers are not required (currently, yes is required).

Convers Cap: This is used by the MQSeries System to determine the translation required for message headers between various hardware platforms on the network. The user data portion of messages is not translated.

Split Mssg: Split or segmented messages not supported at this time.

Other Channel Data

Connection ID: A four-byte field identifying the connection. Required by the sender, optional for the receiver.

Transmission Queue Name: The name of the transmission queue. Required for the sender, optional for the receiver.

TP Name: A sixty four character field identifying the remote task ID of the receiver on the remote CICS region, or a TPNAME on the remote system (for example, MQ03). Required by the sender.

Note: Although the TPNAME may be up to 64 bytes elsewhere, for the MQSeries System purposes it must be up to 4 bytes.

Checkpoint Values

Frequency: Determines checkpoint event based upon I/O frequency.

Time Span: Determines checkpoint event based upon time span in seconds.

Enable: Enable the Dead Letter Queue.

Dead Letter Store: Allow messages for undefined destinations to be written to the Dead Letter Queue.

Modifying and deleting channel definitions

Choice 3 on the configuration menu (the same option as for creating a channel) also allows an operator to modify, or delete channel definitions.

Note: The same primary screen is used in the modify and delete operations as were described above for the add function. This screen is not re-presented here. However, the “LIST” screen is presented and the “flow” for the modify and delete operation is described.

Selecting an existing channel definition

To modify or delete an existing channel definition, the operator must first select the definition on which to work and bring it to the display screen. This can be accomplished by using either of two function keys.

From the “CHANNEL RECORD” screen (this is the first screen displayed after choosing option 3 on the configuration menu), the operator may use either PF4 or PF9.

PF4 is the READ key. It may be used 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. The MQSeries System will read and display the corresponding channel definition.

PF9 is the LIST key. It may be used to bring a specific channel definition to the screen as follows:

1. Press PF9.
2. The MQSeries System will display a list of all defined channels (see screen below).
3. The operator selects the desired channel by typing an “S” next to it.
4. Press PF4, or <**Enter**>.
5. The MQSeries System will read and display the corresponding channel definition.

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:52:01                    Channel List                            VSE2
MQMMCHN                                                             QMMC

S    CHANNEL NAME     TYPE  STATUS LAST MSN  LAST CHECKPOINT

  C1                    R   ENABLE   000009       10:28:31
  MQM7.LU62.VSEF.DEVL   R   ENABLE   005054       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 24. Channel list

On this screen the display fields are:

Channel Name:	The names of all channels.
Type:	Type is Sender or Receiver.
Status:	Channel may be enabled or disabled.
Last MSN:	The last checkpointed message sequence number of the channel.
Last Checkpoint:	The time of the last checkpoint.

Modifying an existing channel definition

Once the desired channel definition has been brought to the display (as described in “Selecting an existing channel definition” on page 82), any field of the definition may be modified just as described in the preceding section for the *add* operation.

When the desired changes have been made, the operator updates the screen via PF6 (=UPDATE).

Deleting an existing channel definition

Once the desired queue definition has been brought to the display (as described in “Selecting an existing channel definition” on page 82), it may be deleted by pressing PF12 (=DELETE). A confirmation request will be displayed, requiring PF12 to be pressed again.

Global system definition display

Choice 4 on the main menu allows an operator to view the attributes defined for the local queue manager (and all system wide parameters) through the following screen:

```
 01/28/1997          IBM MQSeries for VSE/ESA Version 1.4             IYZMZSI2
 11:52:14                  Global System Definition                   VSE2
 MQMMSYS                 Queue Manager Information                    0002
  Queue Manager . . . . . . .: VSE2
  Description Line 1. . . . .: MQ/Series Manager on VSE/ESA 2.1
  Description Line 2. . . . .: Development System
                         Channel Maximum Values
  Checkpointer Global Timer .: 00000060
                         Queue System Values
  Maximum Number of Tasks . .: 00000500      System Wait Interval : 00000030
  Maximum Concurrent Queues .: 00000500      Max. Recovery Tasks  : 0001
  Allow TDQ Write on Errors  : Y   CSMT      Allow Internal Dump  : N
                         Queue Maximum Values
  Maximum Q Depth . . . . . .: 01000000      Maximum Global Locks.: 00000100
  Maximum Message Size. . . .: 00032000      Maximum Local Locks .: 00000100
  Maximum Single Q Access . .: 00000100      Checkpoint Threshold : 1000
                         Global QUEUE /File Names
  Configuration File. . . . .: MQFCNFG
  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
```

Figure 25. Global system definition display

This is a display-only screen.

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

Queue definition display

Choice 5 on the main menu allows an operator to view existing queue definitions.

Note: This function allows an operator to see the queue definition, not the current queue status. To see the current queue information, refer to the Queue Monitor function.

This operation is *identical* to the modify queue and delete queue operations (as described in “Modifying and deleting queue definitions” on page 78) except that the maintenance function keys, PF5(=ADD), PF6(=UPDATE) and PF12(=DELETE), are not available to the operator.

Channel definition display

Choice 6 on the main menu allows an operator to view existing channel definitions.

This operation is *identical* to the modify channel and delete channel operations (as described in “Modifying and deleting channel definitions” on page 82) except that the maintenance function keys, PF6(=UPDATE) and PF12(=DELETE), are not available to the operator.

Operations functions

Selecting option 2 (Operations) from the main menu, causes MQMT to display the following sub-menu screen:

```
01/28/1997        IBM MQSeries for VSE/ESA Version 1.4              IYZMZSI2
11:52:33              *** Operations Main Menu ***                  VSE2
MQMMOPR                                                             0002
                             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.
    5787-ECX (C) Copyright IBM Corp. 1993, 1997  All Rights Reserved.
   ENTER = Process           PF2 = Main Menu             PF3 = Quit
```

Figure 26. Operations main menu

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

Start/Stop queue

Choice 1 on the operations menu allows an operator 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/Stop functions are dynamic with immediate runtime effects. The Get Enabled and Put Enabled functions, on the other hand, are static configuration fields which take effect at system initialization time, or via the Refresh from Config option on this screen. Further, Start/Stop applies universally to all processes attempting to access a local queue, whereas the Get Enabled/Put Enabled flags can be selectively applied to aliases and remote queue definitions.

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:53:25               Start / Stop Queue                           VSE2
MQMMSS                                                              0002

                          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       : GEGE
           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 Queue's  Request
           Function         :        S=Start, X=Stop, or M=Monitor


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

Figure 27. Start/stop queue control screen

On this screen, the fields are:

System Status: Reflects the status of the system. This is normally ACTIVE, unless the system has not been initialized, or unless the system has been shutdown. When this occurs the field will read: 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 unless all queues have been stopped. When this occurs the field will read: 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 unless all channels have been closed. When this occurs the field will read: 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/Stop

Function: The function to be performed.
"S" is to start a stopped local queue, to start the associated trigger mechanism or to start receiving messages if the channel is open.
"X" is to stop a local queue and make it unavailable.
"R" is to refresh the runtime 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. It is important to refresh a queue if its definition is changed, as the change will not otherwise be in 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 then 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 then DISABLED is also displayed.
All Queue's Request	
Function:	This will either stop or start the system queue manager, without effect on system resources. When a queue manager is initiated or shutdown, there are certain bookkeeping functions that must be performed so that the contents of the disk files can be in sync with the storage control blocks. However, when a system is started or stopped, the Queue Manager will simply be enabled or disabled and all resources will be left "as is". The monitor request will toggle the monitor flag. This flag is used to log application requests and their results to the System Monitor Queue.

Notes on the Start/Start Queue panel

Stop/Start - Only local queue definitions can be stopped or started. In order to stop or start a non-local queue (for example, Remote), the queue definition must be updated in the Put-Enabled or Get-Enabled fields. These configuration changes must then be "refreshed" to the runtime environment.

Triggering - When a local queue is started, any associated triggers will also be started, if the Queue Depth reflects that messages are present. This will not happen when a "All Queues Request" function is performed. In addition, any queues that were stopped before the "All Queues Request" stop function was performed, will still be stopped when an "All Queues Request" start function is performed. Use the Monitor Queue function to check which local queues are stopped.

Channel Activation - If the queue definition specifies a trigger and a sender channel, then starting a queue will trigger the sender program to activate the channel and transmit messages.

Open/close channel

Choice 2 on the operations menu allows an operator to open or close communications on an existing channel.

Note: Opening/Closing a Channel is NOT the same as Starting/Stopping the MCA process. See “Communications operations (the MCA process)” on page 96, for further information.

The first screen displayed is:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:54:03               Open / Close Channel                         VSE2
MQMMSC                                                              0002

                          System Information
           System Status    : SYSTEM IS ACTIVE
           Queue Status     : QUEUING SYSTEM IS ACTIVE
           Channel System   : CHANNEL SYSTEM IS ACTIVE
                       Single Channel Request
           Channel Name     : VSE1_TO_VSE2
           Type             :        S=Sender  or  R=Receiver
           Function         : C      O=Open , C=Close , R=Refresh from Config

           Status           :                    DISABLED



                         All Channel's Request
           Function         :        O=Open , C=Close


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

Figure 28. Open/close channel

On this screen the fields are:

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

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

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

Single Channel Request

Channel Name: The name of a specific channel to Start/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 runtime 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, then DISABLED is also displayed.

All Channel’s Request

Function: This will either open or close the channel system.

Note: Opening a channel will not cause a trigger to activate. However, starting the channel’s transmission queue will activate a trigger. See Notes on page 86.

Reset message sequence number

Choice 3 on the operations menu allows an operator to reset the message sequence numbers on an existing channel.

To accomplish this, the screen displayed is:

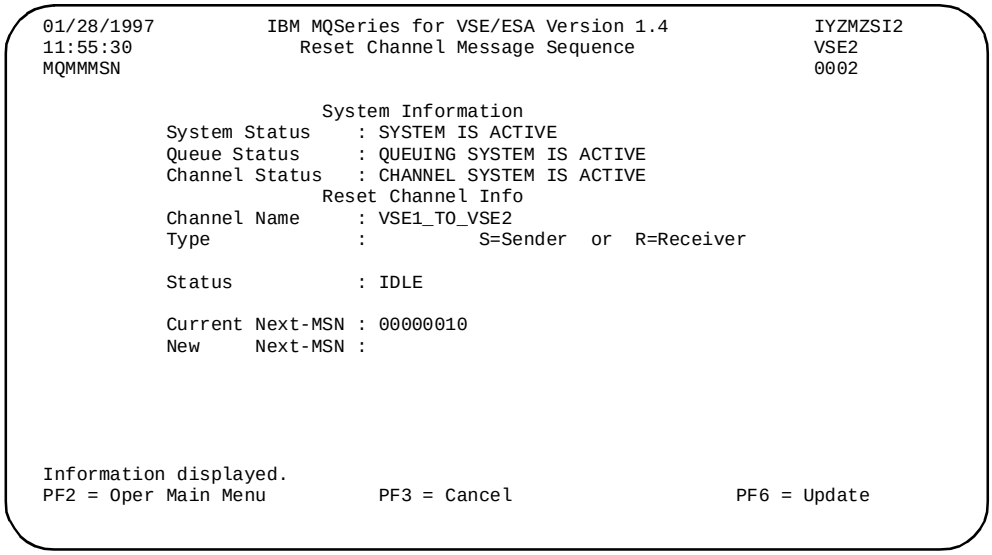


Figure 29. Reset channel message sequence

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 unless the system has been shutdown. When this occurs the field will read: 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 unless all queues have been stopped. When this occurs the field will read: QUEUING SYSTEM IS STOPPED.

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

Reset Channel Info

Channel Name: The name of a specific channel to Open/Close

Status: Reflects the status of the specified channel. This is normally ACTIVE or IDLE unless the channel has been stopped, then DISABLED is displayed.

Current Next-MSN: Displays the message sequence number to be used next.

New Next-MSN: Operator entered field for new 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

Choice 4 on the operations menu allows an operator to initialize the queuing system.

The following screen is displayed:

```
01/28/1997           IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:55:44          Initialization / Shutdown of System                VSE2
MQMMSI                                                               0002

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


            Function         : X        I=Initialize, X=Shutdown


            Returned Results :


             SYSTEM INITIALIZED AT   01/28/199710:28:26



  Please enter one of the options listed.

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

Figure 30. Initialization of system

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 below “Returned Results”.

On this screen the data entry fields are:

Function: Here the system can be initialized or shutdown. If the system is shutdown, this will stop the queue manager and close all channels. If the system is initialized, this will start the Queue Manager and open all channels and queues. Any trigger associated with queues just initialized will also be activated if the Queue Depth is nonzero.

Note: All Queue Maintenance Requests outstanding must have finished before an Initialize or Shutdown operation can be performed.

Queue maintenance

Choice 5 on the operations menu allows the operator to either reset deleted records or physically delete records.

When selected, the following screen is displayed:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
11:56:43            Maintain Queue Message Records                  VSE2
MQMMDEL                                                             0002

                          System Information
            System Status    : System is active.
            Queue  Status    : Queuing system is active.
            Channel System   : Channel system is active.
                       Queue Information
            Queue Name       : GEGE
            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 : 00000015
            Number of Bypass : 00000000
            New Last Read QSN: 00000000
            Process Time     : 00:00:01

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

Figure 31. Maintain Queue Message Records

On this screen the data entry fields are:

Queue Name: The name of the local queue on which the function will be performed.

Function: D = Delete messages that have been logically deleted up to a specified "written" date/time exclusive,

A = Delete all records (logically deleted, or written) and reclaim VSAM space,

R = Reset all logically deleted records to "written" status from a specified "deleted" date/time inclusive.

Note: Specifying D does not actually reclaim VSAM space, because record keys are always created in ascending sequence. It is strongly recommended that the user read "VSAM file maintenance" on page 101 for important information regarding the Delete All function in relation to VSAM files.

Example: Given the date and time of 960930230000, specifying "D" will delete all records with a written time prior to 11:00:00 p.m. Specifying "R" will reset all delivered messages with delivery time after 10:59:59 p.m.

Date: The last date up to which the selected function will be performed (if applicable).

Time: The last time up to which the selected function will be performed (if applicable).

Once the PF6 key is pressed, the function is activated. This function is done by another task which will signal this screen when it is done. This signal can be displayed by pressing the ENTER key. The PF12 key is used only if the Delete task has terminated before finishing the current request. It will act like a new PF6 request.

Notes:

1) A Delete or Reset Messages by Date/Time will perform this function **up to** this Date/Time, but will not include records with this Date/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 will purge all records which include both logically deleted and non-deleted messages.

Once a task to maintain queues is in progress, it flags the Queue Information entry and logically prevents any other task from accessing this queue. Any attempt to open this queue will be 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 main menu, causes MQMT to display the following sub-menu screen:

```
01/28/1997        IBM MQSeries for VSE/ESA Version 1.4              IYZMZSI2
11:58:29             *** Monitor Main Menu ***                      VSE2
MQMMMON                                                             0002
                          SYSTEM IS ACTIVE


                            1. Monitor Queue

                            2. Monitor Channel


                    Option:









Please enter one of the options listed.
    5787-ECX (C) Copyright IBM Corp. 1993, 1997  All Rights Reserved.
  ENTER = Process        PF2 = Main Menu          PF3 = Quit
```

Figure 32. Monitor main menu

On this screen, choices correspond to available system monitor functions.

Monitor queues

Choice 1 on the monitor menu allows an operator to monitor the current status of all existing local queues. The monitor screen displayed is:

```
01/28/1997            IBM MQSeries for VSE/ESA Version 1.4          IYZMZSI2
11:58:35                   Monitor Queues                           VSE2
MQMMMOQ                                                             0002
                           QUEUING SYSTEM IS ACTIVE

QUEUE                        FILE     T INBOUND  OUTBOUND        LR     QDepth
x                            MQFI003  N IDLE     IDLE             0          0
GEGE                         MQFI003  N STOPPED  STOPPED          0          0
LONGQ                        MQFO003  N IDLE     IDLE             0         49
MVS1_LOCAL                   MQFI002  N IDLE     IDLE             0          1
MVS1_TQ                      MQFO003  Y IDLE     IDLE             0          4
QL.DEVL.X                    MQFI001  N IDLE     IDLE             0          4
SIMON                        MQFO003  N IDLE     IDLE             0         10
SYSTEM.EXCEPT                MQFO003  N IDLE     IDLE             0          0
SYSTEM.LOG                   MQFO002  N IDLE     IDLE             0         51
SYSTEM.MONITOR               MQFO002  N IDLE     IDLE             0         14
..More



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

Figure 33. Monitor queues

This screen displays the current status of all local queues. The displayed fields are:

The columns of the display are as follows:

Queue: Name of the queue.

File: CICS FCT DDNAME of a Local Queue definition.

T: Queue type
N - normal local queue
Y - transmit local queue
When PF9 (All) option is selected
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 Put
IDLE - no user has the queue open for Put
STOPPED - queue has been stopped
MAX - at maximum QDepth
FULL - no space
RECOVERY - for dual queuing.

Outbound: Status of the outbound process
ACTIVE - one or more users have the queue open for Get
IDLE - no user has the queue open for Get
STOPPED - queue has been stopped
RECOVERY - for dual queuing.

LR: Last Read: Relative record number of the last record on queue which has been read and processed. (Remember, MQSeries System 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 queue, remaining to be processed.

Note: Est. QDepth is based on all MQPUT requests and only syncpointed MQGET requests.

If the PF9 key is pressed, then an entire list of all queues (local, remote and alias) is displayed with their associated reference. This is a toggle key, if it is pressed again, it will go back to just listing local queues. In this local queue list, a PF10 key will show the detail information for this local queue entry. This information will include trigger and checkpointed information.

Monitor queues - detail

Pressing PF10 will display detail information for a specific channel entry.

The screen displayed is:

```
01/28/1997           IBM MQSeries for VSE/ESA Version 1.4          IYZMZSI2
12:07:05                  Monitor Queues                           VSE2
MQMMMOQ                                                            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 34. Monitor queues - detail

Monitor channel

Choice 2 on the monitor menu allows an operator to monitor the current status of existing communications channels.

The screen displayed is:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
12:10:02                   Monitor Channels                         VSE2
MQMMMOC                                                             0002
                          CHANNEL SYSTEM IS ACTIVE

CHANNEL              TYPE       MSN       QSN      QUEUE
C1                   RECV         9        61 I GEGE
MQM7.LU62.VSEF.DEVL  RECV      5054      3771 I SYSTEM.EXCEPT
MVS1_TO_VSE2         RECV         5        30 I VSE1_LOCAL
VSE1_TO_VSE2         RECV         9         4 C QL.DEVL.X
VSE2_TO_MVS1         SEND        69        69 I MVS1_TQ
VSE2_TO_SD01         SEND         6         6 I TQ.SD01
VSE2_TO_VSE1         SEND         0         0 I









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

Figure 35. Monitor channel definitions

This screen displays the current status of local channels. The displayed fields are:

The columns of the display are as follows:

Channel: Name of the channel.

Type: Sender, Server or Receiver.

MSN: Last Channel Message Sequence Number received or sent.

QSN: Queue Message Sequence Number (of the queue-name displayed in the next field)

QUEUE: Name of the queue associated with the channel. If this is a Receiver channel, then the QUEUE field displays a “snapshot” of the last queue name for which a message was received. This field is preceded by a one character channel status:
I = IDLE
B = BUSY
C = CLOSED (for example, DISABLED).

PF10 will show the detail information for a specific channel entry.

Monitor channel - detail

Pressing PF10 will display detail information for a specific channel entry.

The screen displayed is:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4            IYZMZSI2
12:10:39                   Monitor Channels                         VSE2
MQMMMOC                                                             0002
                          CHANNEL SYSTEM IS ACTIVE

                           DETAIL CHANNEL INFORMATION
VSE1_TO_VSE2
            COMMIT          MSN       QSN DATE/TIME
 RECEIVER    BEFORE           0         0 19970128102830

             AFTER           9         4 19970128102830
                     QL.DEVL.X









Information displayed.
    ENTER = Refresh      PF2 = Main Monitor
                                                 PF10 = List
```

Figure 36. Monitor channel definitions - detail

This screen shows channel activity. It displays the channel name, channel type and the name of the queue it accesses. The MSN, QSN and time stamp of the last commitment for BEFORE COMMIT and AFTER COMMIT are also shown.

Browse function

Selecting option 4 (Browse Queue Records) from the main menu takes the operator directly to a function with no intervening sub-menus.

The screen displayed is:

```
01/28/1997          IBM MQSeries for VSE/ESA Version 1.4             IYZMZSI2
12:11:00                   Browse Queue Records                      VSE2
MQMDISP                      SYSTEM IS ACTIVE                        0002

    Object Name: GEGE
    QSN Number : 00000001       LR-        0, LW-       10, DD-MQFI003
                      Queue Data Record
Record Status : Written.        PUT date/time  : 19970128120243
Record Size   : 00000200        GET date/time  :
Queue line.
THIS IS A MESSAGE TEXT









Information displayed.
    5787-ECX (C) Copyright IBM Corp. 1993, 1997  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 = Monitor
```

Figure 37. Browse queue

This screen shows the content of the message of the message for the specified 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 Queue Sequence Number (QSN) of the message of interest. In the open area on the screen below the Queue Title, the queue message will appear. It can then be manipulated by using the function (PF) keys. Toggling the PF9 key causes the message to be displayed in HEXADECIMAL or EBCDIC text code.

The PF10 key will present detailed MQSeries information for this record. It includes channel information if it is a transmit queue.

If the System Log file is browsed, the PF12 (Help) key will appear and can be used to display User Action and System Action for this message. This function is only available if the runtime system is active.

Note: If the file being browsed is in the process of being updated by any other MQSeries tasks, this function will wait until the completion of those tasks and the user may notice a delay in the response of the browse function.

Communications operations (the MCA process)

The MCA (Message Channel Agent) is the communications engine for the MQSeries System. It runs as a separate CICS task connected to the remote MQSeries System using APPC protocol. The MCA process will automatically start in response to other system activity or when a message is placed on a transmission queue. The operator can control the channels. The MCA process can be stopped from the Operations Main Menu.

Viewing error logs

MQSeries System error messages, and other system informational messages are handled by placing them on queues as follows:

SYSTEM.LOG: All MQSeries System generated error messages are written to this queue. If SYSTEM.LOG is not defined, or if the MQSeries System cannot successfully write to it, then the error messages are logged to CSMT and may be viewed using standard system utilities.[1]

SYSTEM.EXCEPT: Is the MQSeries System Dead Letter Queue. Messages which cannot be properly 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 trouble-shooting purposes.

The names indicated above for these queues are the default names, but the actual queue names may be redefined via the global system definition screen.

The messages written to these queues may be viewed with the MQSeries System browse queue function.

MQSeries command line function

MQCL is a command line interface which allows queues and channels to be selectively opened and closed. It has the flexibility to open and close in inbound, outbound or both directions. The syntax format is:

```
MQCL FF NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
Where:     MQCL       - the Command line transaction ID
           FF         - the function code. Valid codes are
CR  - Close Receiver channel ID    OR  - Open Receiver channel ID
CS  - Close Sender channel ID      OS  - 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
           nnn...nnn  - the name of a Queue or channel
```

A message will be sent to the activated terminal and the SYSTEM.LOG for every activation of this task. This message will have an error code of MQM001000 for completed messages, or an error code of MQM001090 for any ones that did not complete properly. The text that follows will explain the exact results. MQM001090 indicates Command Line Program invalid syntax.

1. This can be changed in the Global System Definition.

Background batch modules

The PRD2.MQSERIES library contains the USER sublibrary. This contains the following example background batch job.

MQJUTILY.Z - Contains the MQPUTIL program which performs the following functions:

1. Prints the system, queues and channels definitions from a configuration file.
2. Prints the SYSTEM.LOG file in a formatted report.
3. Updates all channels with a new starting MSN.
4. Updates a configuration file for dual queues. It will make all dual queues into a primary queue.
5. Print new **Help Facility** error information.

The MQPUTIL program uses the CONFIG DDNAME for the MQSeries System configuration VSAM file, if the “PRINT LOG” command is used. The following is the MQPUTIL program general syntax:

Column	Content
1 to 5	command name
6	space
7 to 18	subcommand
19	space
20...	arguments

Table 32. MQPUTIL program general syntax

MQPUTIL commands:

1. PRINT :
   has 3 subcommands:

CONFIG	Prints the full configuration of the MQSeries System.
LOG	Prints the System Log in a formatted report.
MESSAGES	Prints a HELP Facility resolution Report.

2. RESET :
   has 2 subcommands:

MSN nnnnnn	Resets all channel numbers to nnnnnn passed as argument
CHECKPOINT	Resets all the channels checkpoint values to zero.

   The RESET CHECKPOINT will cause the channel records to be updated with a new checkpoint value. This will cause the current MSN values to be maintained when the MQSeries System runtime system is started. No queue scan is performed to find a later MSN. Essentially, the runtime system is initialized with the last checkpointed MSN for a channel. This is done by using the checkpointed date/time. This value is compared against the updated channel date/time of a queue record. If the queue record is a higher value, then the MSN in the queue record is used in place of the checkpointed value. All of the above implies that if the MQPUTIL program is used to perform the RESET CHECKPOINT function, no queue scan is performed. Whichever checkpoint value was last taken will become the current MSN when the MQSeries System is started.

3. DUALQ :
   has 1 subcommand:

   TAKEOVER dual_queue_name

   Allows the Dual queue specified as argument to become the primary queue.
   The logic is as follows:

   (1).The configuration file will point to the cluster hosting the dual queue instead of to the cluster hosting the primary queue.

(2).All message headers in the dual queue will be modified. They will 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, I/O errors) and a Dual queue has been defined.

It is important to backup the configuration file before using this command, since it will be altered. The configuration file can be restored when the failure is repaired.

The best way to backup this file is by using a VSAM REPRO step.

Using Batch interface

MQSeries/VSE has been designed for online programs only. However in a few cases it might be worth using batch programs as well. For this purpose, 4 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.

COBOL programs are those listed in the Manual “Messaging and Queuing Extensions for VSE/ESA” (GC24-9296). However, they have been slightly modified.

This has been tested in development environments only and is provided on the “as-is” basis. That is, they may have to be modified to fit the user’s environment.

Logic of the Batch Interface

Since only CICS programs may issue MQSeries requests, the idea is to mirror a batch program by a CICS transaction which actually issues the MQSeries Requests. Data Transportation is performed by using XPCC SEND/REPLY protocol.[2]

Because CICS programs cannot use VSE services without a risk of performance degradation, a VSE subtask is attached to the CICS partition to handle all XPCC requests.

Two Assembler programs issue XPCC requests: MQBIBTCH on the Batch side and MQBICITK on the CICS side. So the logic flow is as follows:

```
          Call              XPCC              LINK              Call
program  ----> MQBIBTCH --------> MQBICITK ------> MQBTCIRH -----> MQSeries
```

Indeed, MQseries feedback follows the reverse path.

2. Note for DL/I users: It very similar to the MPS facility.

How to use the Batch Interface

1. In your batch program, issue MQSeries functions the same way you usually do with CICS programs. For example :

```
CALL 'MQCONN'  USING
     QM-NAME-AREA
     HCONN-ADDR-AREA
     CCODE-ADDR-AREA
     RCODE-ADDR-AREA.
```

2. Linkedit 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. Use the transaction MQBI.
4. Execute your batch program.

Note: When your program has terminated, the CICS counterpart does not deactivate. So you may 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 will stop the Batch Interface:

```
ID DIVISION.
PROGRAM-ID.  MQBISTOP.
AUTHOR.      IBM.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
PROCEDURE DIVISION.
CALL 'MQBIEND'.
GOBACK.
```

Data Integrity

Two new functions have been added :

- MQCMIT to commit all changes. This will force a CICS SYNCPOINT be issued by the mirror transaction.
- MQBACK to rollback all changes. The CICS mirror transaction will issue:

```
EXEC CICS SYNCHPOINT ROLLBACK.
```

For both new functions, the syntax is the same:

```
CALL 'funct'   USING
     HCONN-ADDR-AREA
     CCODE-ADDR-AREA
     RCODE-ADDR-AREA.
```

This was to keep the same syntax as for other systems, but none of the passed parameters are actually tested or used.

Under CICS updates are not automatically committed (please refer to “Syncpoints and triggers”, on page 107 for further details), but it is different for batch programs. If a batch program issues the MQDISC call while there are uncommitted requests, an implicit syncpoint occurs.

An implicit rollback occurs, and the Batch interface VSE subtask (under CICS) is terminated if one of the following happens:

a. A MQBIEND call is issued without previous MQDISC call.

b. A system error condition is detected. For example the batch program terminates without issuing a MQDISC call.

Verifying the Batch Interface

The batch program MQBICALL has been provided for this purpose. So, you may use the following job for your first test :

```
// JOB CALLER
// LIBDEF *,SEARCH=(PRD2.MQMSERIES,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 may be running at a time against an MQSeries Queue Manager.
2. Only one CICS partition may run the Batch Interface at a time. However, by changing the Application names in XPPC IDENT you might have multiple versions running. But, still, only one batch program may communicate with one CICS.
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, modifying MQBIBTCH and QMBICIRH is needed.

VSAM file maintenance

All files used by the MQSeries System are VSAM clusters. Most of them 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 52 bytes long, 48 for the queue name and four for the Queue Sequence Number (QSN). 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 occupied is not reclaimed due to the way VSAM works. Therefore, without intervention outside of the MQ manager, there is a high risk of having a VSAM space full condition. This risk is greater when multiple queues share the same physical VSAM cluster (a practice which is allowable, but not recommended).

There are two methods used to reclaim the space of deleted records:

1. By using the online “Delete All” function through the MQMT dialogs.
2. By using the MQPREORG batch program (refer to “MQPREORG function”, on page 102).

Delete all function

Description

In the Maintain Queue Records screen (“Queue maintenance” on page 90), there is a function called “Delete All”. This function will physically delete all messages and reset the QSN to one in order to reclaim freed space. This is a useful tool to maintain the system log file for the MQSeries System. The advantage of this function is that it is an on-line function requiring no other manual operation; simply invoke the function itself.

Warning: Please note that this function will delete **all** messages and should not be used on queue files which contain undelivered messages.

Operation

- Stop the desired queue via the Start/Stop Queue Control screen.
- If the desired queue is a transmission queue, stop only the inbound direction first. When the queue depth reaches zero, then stop the outbound and close the associated Sender channel.
- If the desired queue is a destination queue with trigger capability, close the associated Receiver channel.
- Enter the Queue Name with “A” for function in Maintain Queue Records screen and press PF6 for update.
- Press enter for result.
- After “Queue Processing Finished” is displayed, start the reorganized queue in the Start/Stop Queue Control screen.

MQPREORG function

The MQSeries System distribution includes a batch program utility called MQPREORG and sample JCL to run MQPREORG.

Description

This utility is designed to be used as a nightly or weekly queue cleanup facility. Either every queue or only one queue file can be reorganized in a job step. This function accepts the queue name from SYSIPT and the name of the VSAM file from DLBL. All messages are bypassed except the messages marked as “Written” (to be delivered) in the specified queue. The retained “Written” messages are resequenced and written into 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 are to be retained, a simple ‘delete-and-define’ IDCAMS JCL is sufficient for the job.

Multiple queues sharing a VSAM cluster

If there is more than one queue defined in a VSAM cluster, then all queues have to be processed before deleting and recreating this cluster. Otherwise, records from unprocessed queues would be lost. To help the user reorganize all queues, he may use the “ALL” option instead of the queue name.

```
// EXEC MQPREORG
ALL
/*
```

In fact, in most cases, “ALL is the only option that will be used. Specifying queue names is only worthwhile when the user wants to move a queue from one cluster to another.

Reorganizing queue files while the queue manager is down

- Procedure:

1. If CICS is up, use CEMT to disable and close the VSAM file(s) to be processed.
2. Modify the sample JCL to include your system parameters and reorganization requirements. Then execute the job to run the batch program utility, MQPREORG, to reorganize the VSAM file(s) and reclaim all freed space.
3. If Step 1 was performed, use CEMT to open and enable the processed VSAM file(s).

Sample JCL to run MQPREORG

```
* ** JOB JNM=MQJREORG,DISP=D,CLASS=O
* ** 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 ?volid? 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 reside 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 MQIFOO2. Changes must be done for other clusters.   *
* ------------------------------------------------------------------*
* Licensed Materials - Property of IBM                               *
*                                                                    *
* 5787-ECX                                                           *
* (C) Copyright IBM Corp. 1993, 1996                                 *
*                                                                    *
* US Government Users Restricted Rights - Use, duplication or        *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp.  *
* ------------------------------------------------------------------*
// DLBL INPUTQ,'MQSERIES.MQFIOO2',,VSAM,CAT=MQMCAT
// DLBL OUTPUTQ,'MQSERIES.WORK.QUEUE',,VSAM,CAT=MQMCAT
// EXEC IDCAMS,SIZE=AUTO
 /*                                                                 */
 /*                   VERIFY VSAM FILE                              */
 /*                                                                 */
          VERIFY FILE(INPUTQ)
          IF MAXCC > O 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 (?volid?)                         -
                    NONINDEXED)                               -
                 DATA                                         -
                   (NAME (MQSERIES.WORK.QUEUE.DATA)           -
                    RECORDSIZE (2048 32048)                   -
                    CISZ (8096))                              -
                    CAT (?CAT?)
/*
// 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 7. Application programming interface

The MQSeries System application programming interface implements the IBM Message Queue Interface (MQI). This simple set of calls provides a way for applications to exchange messages with other MQSeries Systems such as MVS/ESA, RISC System/6000, VAX, TANDEM, AS/400, PCs, etc.

The applications programmer/analyst/designer should read earlier chapters of this document for an overall understanding of the MQSeries System.

In addition to these sources, this chapter provides:

- General information regarding the MQI
- Design guidelines for applications wishing to use the MQI
- Detailed reference information for each individual MQI function
- Descriptions of key MQI data structures
- Completion codes and reason codes returned by MQI functions

Working with the MQI

The MQI is responsible for handling user application requests to read and write from the queuing system, and for arbitrating among multiple requests to the same queue.

In IBM MQSeries for VSE/ESA, the MQI is built around the standard COBOL language function call interface which allows a fixed number of arguments.

MQI calls and sequence of operations

The MQI calls supported by IBM MQSeries for VSE/ESA are:

MQCONN	Connects the application to the MQSeries System Queue Manager
MQOPEN	Opens access to a specific queue
MQGET	Reads a message from a specified queue
MQPUT	Writes a message to a specified queue
MQPUT1	Opens a queue, writes one message, and closes the queue
MQINQ	Inquires about queue status information
MQCLOSE	Closes access to a specific queue
MQDISC	Disconnects the application from the MQSeries System queue manager

These calls are described in detail in “MQI calls reference” on page 112. It is also important to understand the data structures required by the interface -- especially as part of the `MQGET` and `MQPUT` calls. The primary structures are:

MQMD	MQ Message Descriptor
MQGMO	MQGet Message Options
MQPMO	MQPut Message Options
MQOD	MQ Object Descriptions

The use of these data structures is described along with the MQI call descriptions in “MQI calls reference” on page 112. The structures are described independently in “MQI data types and structures” on page 130.

The sequence of MQI operations performed by an application is very similar to the sequence used for any familiar record-oriented I/O subsystem. That is, just as one must OPEN and CLOSE a disk file, one must connect to and open (`MQOPEN`) a queue before accessing it, and must close (`MQCLOSE`) and then disconnect at the completion of processing. Within the application, the user requirements will determine the sequence of `MQGET` and `MQPUT` operations.

Sample source code provided

One sample trigger program, MQPECHO is provided with IBM MQSeries for VSE/ESA. The source code for this program can be found in the Appendix, or it can be listed directly from the distribution files.

Within the source code for MQPECHO, the user will find examples which illustrate the use of the MQI calls in a trigger program.

In addition there are three sample programs, TTPTST1, TTPTST2 and TTPTST3 plus COBOL language copybook files which are provided with the distribution in the **PRD2.MQSERIES** library. These files provide examples of all of the MQI calls.

Compiling your application program

The MQI verbs are provided in the library PRD2.MQSERIES.

Compilation

Make sure to include the PRD2.MQSERIES library as part of the application phase step.

Applications not written in COBOL for VSE

For CICS, COBOL is the language in which the MQI is written. Applications written in COBOL for VSE have been thoroughly tested with the MQI. Sample programs and copybooks are provided in COBOL for VSE.

COBOL for VSE is clearly the language of choice for development of the MQSeries System applications on CICS. However, for a variety of reasons, some users will want to write in another programming language.

In these cases, the *customer* must meet the interface requirements of the COBOL language interface. There are no sample programs and no includable copybook files provided in any other language.

Nevertheless, any programming language which can call COBOL routines should be able to be used in one of two manners:

- Call the MQI directly from another language. This usage requires that all interface parameters match up identically at the binary level. With some languages this may present a problem. For example, it is NOT possible to write programs calling the MQSeries API in Assembler language since there is no support for LE/VSE conforming assembler main routines under CICS. Please refer to “LE/VSE Programming Guide” for further details.

or

- Within the application, call an application subroutine written in COBOL for VSE. From this COBOL language subroutine issue static MQI calls. In this manner, there should be no problem with data alignment.

Application design guidelines

The hidden network

One of the key benefits provided by the MQI is the ability for a distributed application to be developed which is totally independent of the underlying network. This network independence means there is no need for an application to be aware of *either*:

- The lower levels of the communication protocol(s)

or

- The physical location of other applications on the network.

In order to take full advantage of this network independence, the queue names used by the application must be chosen properly.

In particular, it is recommended that application programs use only a *single logical name* to refer to each MQSeries System queue. For the MQI 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 reasoning behind this, parallels the logical naming used in other I/O subsystems. When dealing with disk subsystems, no application hard-codes the device name and path name for a file. This would cause problems for the application when normal system management functions relocate a file.

The same is true when addressing MQSeries System queues. Since the *Queue_Manager_Name* is typically associated with a particular system, its use implies knowledge of the physical network. This can place restrictions on any future modifications to the network and increase the probability that network changes will require changes to the source code of applications.

Note: The use of the Queue_Name field as the only logical queue name is strongly recommended. 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.

This recommended usage should be reflected in the list of queue names defined by the system designer (as described in Chapter 3, “Planning” on page 15).

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 are 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 109.

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 take effect. For some applications, queues need to be coordinated. Applications need to be able to get and put messages (and possibly update other resources, such as databases), and know that either all of the operations take effect, or that none of the operations take effect. This set of coordinated operations is called a *unit of work.* An example of a unit of work would be a debit and credit for a funds transfer in a financial application. Both operations must complete, or neither operations 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.

When a *syncpoint is declared*, any party (applications and a 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 execute CICS SYNCPOINT prior to invoking a 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 putting system, 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, between the completion of the MQPUT call[1] and the successful completion of the unit of work, the message is not available to be retrieved from the target queue, except from 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 big 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, then 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

Currently only persistent messaging is supported. Persistent messages do not get deleted if the Queue Manager is restarted. Thus, they are fully recovered when the Queue Manager is restarted. Syncpointing by the application will cause these records to be in a logical unit of work. Therefore, any records that were syncpointed will still be recovered if the Queue Manager is shutdown and restarted.

---

1. In this discussion it is assumed that call completes with MQCC_OK.

Syncpoint Rollback

If your application wants to undo what has been done since the beginning of the current LUW, it has to issue:

```
EXEC CICS SYNCPOINT ROLLBACK
```

This might have the following (and non-desired) results:

- Monitoring will show wrong queue depth values. This is because the Queue Manager is not aware of rollbacks. Unless having a logic which would drastically impact the overall performance, it was no satisfying solution to overcome this problem. This value is correctly reset when stopping, then 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) will never be used again. This is because other applications may have also put messages into the same queue. Let's take a simple example:

  Transaction A writes Message number 5
  Transaction B 6
  Transaction A 7
  Transaction C 8

  At this point the queue depth is 8. Now let's assume Transaction A rolls back, then Messages 5 and 7 will be never retrieved (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 should have no impact at all, but at first glance might 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, defining 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, having 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 when the volume of message traffic fluctuates wildly. 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 trigger facility

A local queue definition can have a trigger event associated with it when it is defined. This event is defined to activate the MQM Trigger API Handler (i.e., MQ02 CICS Transaction). The Trigger API Handler will do either a CICS LINK to the application program or a CICS START to the application transaction. This is based on whether the user 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 by using the COMMAREA.
   To return to the API handler, the user should issue an EXEC CICS RETURN.
2. If the trigger facility specified a transaction name, this information area can be gotten by issuing an EXEC CICS RETRIEVE command.
   Before exiting from the program, the user must issue a 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 will recognize this transaction ID for a local queue being opened. When this queue is closed fully, then this trigger event will be closed, thus allowing another trigger for this queue to be activated.

Trigger conditions

The queue manager will activate a trigger event based on the event type defined for the current queue against which the MQPUT operation has been requested.

Note: The trigger condition suffices automatically if a non-empty queue is stopped then restarted, regardless of the trigger event type.

The Trigger API handler will wait until this MQPUT request has been completed. This implies that the MQPUT could have been successful or unsuccessful (that is, rolled back). The activated trigger application program should perform an MQGET call. If the result of this MQGET is an empty condition (that is, MQRC_NO_MSG_AVAILABLE), then 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 terminate.

A trigger event type of “FIRST” will generate a trigger event to be performed 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 of “EVERY” will generate a trigger event to be performed after every MQPUT has been completed, up to the maximum number of trigger events specified on the Extended Local Queue Configuration Screen.

Defining a sender channel component

A sender channel component will cause the channel to be started if there are messages on the transmission queue to be sent to the remote node. (A server channel component, by contrast, 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 does not support requester component.

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 trigger facility” on page 109 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 trigger facility" on page 109, for CICS START, provides details of how to trigger a program based on its transaction ID. Note that the transaction should not also be invoked outside the trigger mechanism; however, by defining a different transaction name with the same program name, the program may be invoked outside of 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>

Message batch processing

Message batch processing is not supported on IBM MQSeries for VSE/ESA.

MQI calls reference

For each of the MQI functions, this section presents the detailed *call format*, *parameters*, and *guidelines* in the following format:

The API calls are described using the following conventions:

> ### MQAPINAME - Call Name
>
> `CALL 'MQAPINAME' using Parameter1 Parameter2 Parameter3 ... ParameterN.`
>
> Description of how and when to use the API call.
>
> ### Parameters
>
> ***Parameter 1 - Parameter Type (see below)***
>
> Description of Parameter.
>
> Instructions for using the parameter.
>
> - PARAMETER_OPTION - description
> - PARAMETER_OPTION_2 - description.
>
> ### Guidelines
>
> Guidelines and tips for using the call.

Parameters :

Parameters are variables having particular data types or mapping particular data stuctures.The name of the associated data type, or structure is shown between parenthesis. Refer to “MQI data types and structures” on page 130 for detailed information.

Parameter types:

- **Input** - Parameter set by the application for use by the queue manager.
- **Output** - Parameter set by the queue manager for use by the application on return from the call.
- **Input/Output** - Set by the application for use by the queue manager, and modified by the queue manager for use by the application on return from the call.

Notes: Features of the full MQI that are not supported in IBM MQSeries for VSE/ESA are so noted in the following sections.
API functions must be called by using “STATIC CALLS”, otherwise program abends will likely occur. Static calls implies that the module name being enclosed between quotes. Example:

```
CALL 'MQOPEN' USING ........
```

MQCONN - connect queue manager

```
CALL 'MQCONN' USING Name
        Hconn
        CompCode
        Reason
END-CALL.
```

The MQCONN call connects an application program to a queue manager. It provides a queue manager handle, which is used by the application on subsequent message-queuing calls.

Before any of the message-queuing services can be used, the application must establish a connection to a queue manager. The application does this by means of the MQCONN call.

The application provides the name of the queue manager required (Name), and receives in return a handle (Hconn) that represents the connection to that queue manager. Only the Local Queue Manager can be used. No substitution via an Alias Queue Manager can be used.

The returned handle is needed for all subsequent calls on that connection.

CompCode and Reason are returned parameters that indicate the success or failure of the call.

The application can connect either to a specified queue manager, or to the default queue manager.

The default queue manager is requested by specifying a name consisting entirely of blanks or null bytes. The queue manager specified must be local to the application.

Parameters

Name (MQCHAR48) - input

Name of the queue manager.

The name specified must be the name of the local queue manager; if the name consists entirely of blanks, the name of the default queue manager is used (in VSE, there is only one queue manager in a CICS partition).

The name must not contain leading or embedded blanks, but may contain trailing blanks or null bytes; the first null character and characters following it are treated as blanks.

Hconn (MQHCONN) - output

Connection handle.

This handle represents the connection to the queue manager. It must be specified on all subsequent message-queuing call issued by the application. It ceases to be valid when the MQDISC call is issued, or when the application ends.

CompCode (MQLONG) - output

Completion code.

It is one of the following:

- MQCC_OK - Successful completion.
- MQCC_WARNING - Warning (partial completion).
- MQCC_FAILED - Call failed.

Reason (MQLONG) - output

Reason code qualifying CompCode.

If CompCode is MQCC_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_WARNING:

- MQRC_ALREADY_CONNECTED - Application already connected.

If CompCode is MQCC_FAILED:

- MQRC_MAX_CONNS_LIMIT_REACHED - Maximum number of connections reached.
- MQRC_Q_MGR_NAME_ERROR - Queue manager name is not Local Queue Manager Name.
- MQRC_Q_MGR_NOT_AVAILABLE - Queue manager not available for connection.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.
- MQRC_UNEXPECTED_ERROR - Unexpected error occurred.

See “MQI return codes” on page 141, for more details.

Guidelines

1. Only a local queue manager can be connected using this call; it is not possible to connect to a remote queue manager. Queues which belong to the connected queue manager appear to the application as local queues. Queues belonging to local queue managers other than the connected queue manager appear as remote queues. Queues belonging to remote queue managers also appear as remote queues.
2. After a failure of a connection to the queue manager, this call must be reissued. The application program can keep reissuing MQCONN calls until it finds that the queue manager has been restarted. If an application is not sure whether or not it is connected to the queue manager, it can safely reissue an MQCONN call. If it is already connected, the same handle is returned as was returned for the previous MQCONN call.
3. The MQDISC call is used to disconnect from the queue manager.

MQOPEN - open message queue

```
CALL 'MQOPEN' USING Hconn
        ObjDesc
        Options
        Hobj
        CompCode
        Reason
END-CALL.
```

The MQOPEN call establishes access to a queue object.

When a connection to the queue manager has been established, the application can open one or more queues for putting or getting messages. A queue is opened by means of the MQOPEN call.

The application specifies the queue to be opened (ObjDesc), and options (Options) that indicate whether the queue is opened for putting or getting messages.

The application receives in return a handle (Hobj) to the opened queue. The returned handle is used on subsequent calls to access the queue.

Parameters

Hconn (MQHCONN) - input

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

ObjDesc (MQOD) - input/output

Object descriptor.

This is the structure that identifies the object to be opened; see MQOD in “MQOD - MQ object descriptor structure” on page 132, for details.

Options (MQLONG) - input

Options that control the action of the MQOPEN call.

One or more of the following must be specified. If more than one is required, the values are added together.[2] Combinations that are not valid are noted; all other combinations are valid. Only options that are applicable to the type of object specified by ObjDesc are allowed.

2. Do not add the same constant more than once.

The options for controlling the action of MQOPEN are as follows:

- Only one of MQOO_INPUT_SHARED and MQOO_INPUT_EXCLUSIVE options can be specified.
- MQOO_INPUT_SHARED - Open to get messages with shared access. The queue is opened for use with subsequent MQGET calls. The call can succeed if this queue is currently open, by this or another application, with MQOO_INPUT_SHARED, but fails if it is currently open with MQOO_INPUT_EXCLUSIVE.
- MQOO_INPUT_EXCLUSIVE - Open to get messages with exclusive access. The queue is opened for use with subsequent MQGET calls. The call fails if this queue is currently open, by this or another application, for input of any type (MQOO_INPUT_SHARED or MQOO_INPUT_EXCLUSIVE). Only one of MQOO_INPUT_SHARED and MQOO_INPUT_EXCLUSIVE options can be specified.
- MQOO_BROWSE - Open to browse messages. The queue is opened for use with subsequent MQGET calls with the MQGMO_BROWSE_FIRST or MQGMO_BROWSE_NEXT option. This is allowed even if the queue is currently open for MQOO_INPUT_EXCLUSIVE. An MQOPEN call with the MQOO_BROWSE option establishes a browse cursor, and positions it logically before the first message on the queue.
- MQOO_OUTPUT - Open to put messages. The queue is opened for use with subsequent MQPUT calls.
- MQOO_INQUIRE Open to inquire object attributes. The queue is opened for use with subsequent MQINQ calls.

Notes:

1. MQOO_OUTPUT cannot be used with an input option(s) (e.g. MQOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE, or MQOO_BROWSE). Another MQOPEN must be used instead.
2. Queue(s) opened with MQOO_BROWSE and/or MQOO_INQUIRE options only will not affect the Queue Outbound Status on the Monitor Queue Definition Screen.

Table 33. Valid open options for each queue type

Option	Alias[a]	Local	Remote
MQOO_INPUT_SHARED	X	X	
MQOO_INPUT_EXCLUSIVE	X	X	
MQOO_BROWSE	X	X	
MQOO_OUTPUT	X	X	X
MQOO_INQUIRE	X	X	X

a. The validity of an alias depends on the validity of the queue to which the alias resolves.

If an alias queue is being opened for input (browse does not count as input), the test for exclusive use (or for whether another application has exclusive use) is against the base queue to which the alias queue resolves.

Hobj (MQHOBJ) - output

Object handle.

This handle represents the access that has been established to the object. It must be specified on subsequent message-queuing calls, such as MQGET, MQINQ and MQPUT, that operate on the object. It ceases to be valid when the MQCLOSE call is issued, or when the application ends.

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_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_FAILED:

- MQRC_ALIAS_BASE_Q_TYPE_ERROR - Alias base queue not a valid type.
- MQRC_CONNECTION_BROKEN - Connection lost.
- MQRC_HANDLE_NOT_AVAILABLE - No more handles available.
- MQRC_HCONN_ERROR - Connection handle not valid.
- MQRC_OBJECT_IN_USE - Object already open with conflicting options.
- MQRC_OBJECT_TYPE_ERROR – Object type not valid.
- MQRC_OD_ERROR - Object descriptor structure not valid.
- MQRC_OPTION_NOT_VALID_FOR_TYPE - Options not valid for object type.
- MQRC_OPTIONS_ERROR - Options not valid or not consistent.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.
- MQRC_UNEXPECTED_ERROR - Unexpected error occurred.
- MQRC_UNKNOWN_ALIAS_BASE_Q - Unknown alias base queue.
- MQRC_UNKNOWN_OBJECT_NAME - Unknown object name.
- MQRC_UNKNOWN_OBJECT_Q_MGR - Unknown object queue manager.
- MQRC_UNKNOWN_REMOTE_Q_MGR - Unknown remote queue manager.

See “MQI return codes” on page 141, for more details

Guidelines

1. This call is used to open a queue in order to:
   - Get messages (using MQGET call).
   - Put messages (using the MQPUT call).
   - Inquire about the attributes of the queue (using the MQINQ call).
2. It is valid for an application to open the same object more than once. Each handle that is returned can be used for the functions for which the corresponding open was performed.
3. All name resolution within the local queue manager instance takes place at the time of the MQOPEN call. This may include one or more of the following for a given MQOPEN call:
   - Alias resolution to base queue name.
   - Resolution of remote queue name to remote queue manager name, and the local queue name by which it is known at the remote queue manager.

   However, be aware that subsequent MQINQ calls for the handle relate solely to the name that has been opened, and not to the object resulting after name resolution has occurred. For example, if the object opened is an alias, the attributes returned by the MQINQ call are the attributes of the alias, not the attributes of the base queue to which the alias resolves.
4. The attributes of an object (including the result of name resolution) can change while an application has the object open.
5. A remote queue can be specified in one of two ways in the ObjDesc parameter of this call (see the ObjectName field in “MQOD - MQ object descriptor structure” on page 132).

- By specifying ObjectName as the local resource-name of the remote queue, as known to the local queue manager. In this case, ObjectQMgrName refers to the connected queue manager. See “Queue name format” on page 24, for details.
- By specifying ObjectName as the local resource-name of the remote queue, as known to the remote queue manager. In this case, ObjectQMgrName is the name of the remote queue manager.

In either case:

- No message flows occur at the time of an MQOPEN call to the remote queue manager to perform authorization checks.

6. An MQOPEN call with the MQOO_BROWSE option establishes a browse cursor, for use with the MQGET calls that specify the object handle and one of the browse options. This allows the queue to be scanned without altering its contents. A message that has been found by browsing can subsequently be removed from the queue using the MQGMO_MSG_UNDER_CURSOR option.

   Each established browse cursor adversely impacts the performance of non-browse MQGET calls. It is recommended therefore that browse operations should be completed as rapidly as possible, and the cursor destroyed by closing the queue. If further browse operations are required later, it is better to close the queue and reopen it when needed, in order to establish a new browse cursor.

   Multiple browse cursors can be active for a single application issuing several MQOPEN requests for the same queue.

MQGET - get message

```
CALL ‘MQGET’ USING Hconn
        Hobj
        MQMD
        GetMsgOpts
        BufferLength
        Buffer
        DataLength
        CompCode
        Reason
END-CALL.
```

The MQGET call retrieves a message from a local queue that has been opened using an MQOPEN call.

For a queue that has been opened for getting, the application can get messages from that queue by means of the MQGET call.

The application specifies a partially filled-in message descriptor (MsgDesc), some options that control the action of the call (GetMsgOpts), an empty buffer (Buffer), and the length of the buffer (BufferLength).

The application receives in return the message data in the buffer (Buffer), and the total length of the message data (DataLength). The message descriptor (MsgDesc) is completed with information about the message just retrieved.

The MQGET call can be used repeatedly to get many messages from the same queue, without the intervening use of the MQOPEN and MQCLOSE calls.

Parameters

Hconn (MQHCONN) - input

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

Hobj (MQHOBJ) - input

Object handle.

This handle represents the queue from which a message is to be read. The queue must have been opened with one or more of the following options (see the MQOPEN call for details):

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_BROWSE

MsgDesc (MQMD) - input/output

Message descriptor.

This structure describes the attributes of the message required, and the attributes of the message retrieved. See MQMD, “MQMD - MQ message descriptor structure” on page 133, for the format of the message descriptor.

If BufferLength is less than the message length, MsgDesc is still filled in by the queue manager, whether or not MQGMO_ACCEPT_TRUNCATED_MSG is specified on the GetMsgOpts parameter (see the Options field in “MQGMO - MQ get message options structure” on page 139, for more information).

GetMsgOpts (PMQGMO) - input/output

Options that control the action of an MQGET call.

See MQGMO in “MQGMO - MQ get message options structure” on page 139, for details.

BufferLength (MQLONG) - input

Length in bytes of the Buffer area.

Maximum message size is defined by the access queue manager parameters.

Buffer (MQBYTExBufferLength) - output

Area to contain the message data.

If BufferLength is defined as less than the message length, as much of the message as possible is moved into Buffer, whether or not MQGMO_ACCEPT_TRUNCATED_MSG is specified on the GetMsgOpts parameter.

If character data is used within the application message text, the coded character set identifier has to be agreed between the sending and receiving applications, or else the character set has to be limited to the subset that is known to occupy the same code points for both the sender and receiver.

DataLength (MQLONG) - output

Length of the message.

This is the length of the application data in the message. If this is greater than BufferLength, only BufferLength bytes are returned in the Buffer parameter (the message is truncated). If the value is zero, it means that the message contains no application data.

If BufferLength is less than the message length, DataLength is still filled in by the queue manager, whether or not MQGMO_ACCEPT_TRUNCATED_MSG is specified on the GetMsgOpts parameter (see the Options field in “MQGMO - MQ get message options structure” on page 139, for more information). This allows the application to determine the size of the buffer required to accommodate the message data, and then reissue the call with a buffer of the appropriate size.

CompCode (MQLONG) - output

Completion code.

It is one of the following:

- MQCC_OK - Successful completion.
- MQCC_WARNING - Warning (partial completion).
- MQCC_FAILED - Call failed.

Reason (MQLONG) - output

Reason code qualifying CompCode.

If CompCode is MQCC_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_WARNING:

- MQRC_TRUNCATED_MSG_ACCEPTED - Truncated message returned (message deleted from queue).
- MQRC_TRUNCATED_MSG_FAILED - Truncated message returned (message not deleted from queue).

If CompCode is MQCC_FAILED:

- MQRC_BUFFER_LENGTH_ERROR - Buffer length parameter not valid.
- MQRC_CONNECTION_BROKEN - Connection lost.
- MQRC_CORREL_ID_ERROR - CorrelId must be LOW-VALUES
- MQRC_FILE_SYSTEM_ERROR - Queuer received file error. See system log for details.
- MQRC_GET_INHIBITED - Gets inhibited for the queue.
- MQRC_GMO_ERROR - Get options are invalid.
- MQRC_HCONN_ERROR - Connection handle not valid.
- MQRC_HOBJ_ERROR - Object handle not valid.
- MQRC_MD_ERROR - Message descriptor not valid.
- MQRC_MSG_ID_ERROR - MsgId must be LOW-VALUES
- MQRC_NO_MSG_AVAILABLE - No message available to satisfy specified operation.
- MQRC_NO_MSG_ID_ERROR - An unlock request was rejected.
- MQRC_NO_MSG_UNDER_CURSOR - Browse cursor not positioned on message.
- MQRC_NOT_OPEN_FOR_BROWSE - Queue object not open for browse.
- MQRC_NOT_OPEN_FOR_INPUT - Queue object not open for input.
- MQRC_OPTIONS_ERROR - Options not valid or consistent.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.
- MQRC_WAIT_INTERVAL_ERROR - Negative wait interval in MQGMO.

See “MQI return codes” on page 141, for more details

Guidelines

1. The message retrieved is normally deleted from the queue as part of the MQGET call. Message deletion does not occur if an MQGMO_BROWSE_FIRST or MQGMO_BROWSE_NEXT option is specified on the GetMsgOpts parameter.
2. Applications should look for the feedback code MQFB_QUIT (on the MsgDesc parameter) and end if they get such a message - see the Feedback field for more information.

MQPUT - put message

```
CALL 'MQPUT' USING Hconn
        Hobj
        MsgDesc
        PutMsgOpts
        BufferLength
        Buffer
        CompCode
        Reason
END-CALL.
```

The MQPUT call puts a message on a queue; the queue must already be open.

When the queue has been opened for putting, the application can put messages to that queue by means of the MQPUT call.

The application specifies information about the message to be put (MsgDesc), options that control the action of the put (PutMsgOpts), the length of the data (BufferLength), and the message itself (Buffer).

The MQPUT call can be used repeatedly to put many messages on the same queue, without intervening use of the MQOPEN and MQCLOSE calls.

Parameters

Hconn (MQHCONN) - input

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

Hobj (MQHOBJ) - input

Object handle.

This handle represents the queue to which the message is added. The queue must be opened for MQOO_OUTPUT (see the MQOPEN call).

MsgDesc (MQMD) - input/output

Message descriptor.

This structure describes the attributes of the message being sent, and receives feedback information after the put request is complete. See MQMD in “MQMD - MQ message descriptor structure” on page 133, for the format of the message descriptor.

PutMsgOpts (MQPMO) - input/output

Options that control the action of the MQPUT call.

See MQPMO in “MQI data types and structures” on page 130, for details.

BufferLength (MQLONG) - input

Length of the message in Buffer.

Zero is valid, and indicates that the message contains no application data.

Buffer (MQBYTExBufferLength) - input

This is a buffer containing the application data to be sent.

CompCode (MQLONG) - output

Completion Code.

It is one of the following:

- MQCC_OK - Successful completion.
- MQCC_FAILED - Call failed.
- MQCC_WARNING - Warning (partial completion).

Reason (MQLONG) - output

Reason code qualifying CompCode.

If CompCode is MQCC_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_WARNING:

- MQRC_PRIORTY_EXCEEDS_MAXIMUM - A priority > 0 was specified and is ignored by MQSeries System.

If CompCode is MQCC_FAILED:

- MQRC_BUFFER_LENGTH_ERROR - Buffer length parameter not valid.
- MQRC_CONNECTION_BROKEN - Connection lost.
- MQRC_EXPIRY_ERROR - Expiry time not valid.
- MQRC_FEEDBACK_ERROR - Feedback code not valid.
- MQRC_HCONN_ERROR - Connection handle not valid.
- MQRC_HOBJ_ERROR - Object handle not valid.
- MQRC_MD_ERROR - Message descriptor not valid.
- MQRC_MISSING_REPLY_TO_Q - Missing reply-to-queue.
- MQRC_MSG_TOO_BIG_FOR_Q - Message length greater than maximum for queue.
- MQRC_MSG_TYPE_ERROR - Message type in message descriptor not valid.
- MQRC_NOT_OPEN_FOR_OUTPUT - Queue object not open for output.
- MQRC_OPTIONS_ERROR - Options not valid or not consistent.
- MQRC_PERSISTENCE_ERROR - Persistence not valid.
- MQRC_PMO_ERROR - Put-message-options structure not valid.
- MQRC_PRIORITY_ERROR - Priority not valid.
- MQRC_PUT_INHIBITED - Puts inhibited for queue.
- MQRC_Q_FULL - Queue already at maximum depth.
- MQRC_Q_SPACE_NOT_AVAILABLE - No space available on disk for queue.
- MQRC_REPORT_OPTIONS_ERROR - Report options in message descriptor not valid.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.
- MQRC_UNEXPECTED_ERROR - Unexpected error occurred.

See “MQI return codes” on page 141, for more details.

Guidelines

1. The MQPUT call should be used when multiple messages are to be placed on a queue. An MQOPEN call, with the MQOO_OUTPUT attribute, is first issued, followed by one or more MQPUT requests to add messages to the queue. The queue is then closed with an MQCLOSE call.
2. If only one message is to be put on the queue, the MQPUT1 call can be used.

MQCLOSE - close object

```
CALL 'MQCLOSE' USING Hconn
Hobj
Options
CompCode
Reason
END-CALL.
```

The MQCLOSE call relinquishes access to an object, and is the inverse of the MQOPEN call.

When the application has finished putting messages on a queue, or getting messages from a queue, the application must close the queue by means of the MQCLOSE call.

The application specifies the handle of the queue to be closed (Hobj), and some options that control the action of the call (Options). After the call, the queue handle (Hobj) is no longer valid, and messages cannot be put to the queue or removed from the application unless it performs another MQOPEN call.

An application that is reading from a queue does not have to empty the queue before closing it. Messages left on a queue are retained by the queue manager, and may be accessed later by the same or another application.

Parameters

Hconn (MQHCONN) - input

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

Hobj(MQHOBJ) - input/output

Object Handle.

This handle represents the object which is being closed. The value of Hobj was returned by a previous MQOPEN call.

On successful completion of the call, the queue manager sets this parameter to a value that is not a valid handle.

Options (MQLONG) - input

Options that control the action of an MQCLOSE call.

The following must be specified:

MQCO_NONE - No optional close processing required.

CompCode (MQLONG) - input

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_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_FAILED:

- MQRC_CONNECTION_BROKEN - Connection lost.
- MQRC_HCONN_ERROR - Connection handle not valid.
- MQRC_HOBJ_ERROR - Object handle not valid.
- MQRC_OPTIONS_ERROR - Option(s) are not valid.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.

See “MQI return codes” on page 141, for more details

Guidelines

1. When an application issues the MQDISC call, or ends either normally or abnormally, any objects which were opened by the application and which are still open are closed automatically with the MQCO_NONE option.
2. If operations on a queue were performed under syncpoint control, the queue can be closed before or after the syncpoint occurs without affecting the outcome of the syncpoint.

If the queue was opened with the MQOO_BROWSE option, the browse cursor is destroyed by MQCLOSE. If the queue is subsequently reopened with the MQOO_BROWSE option, a new browse cursor is created (see the MQOO_BROWSE option in “MQI data types and structures” on page 130).

MQDISC - disconnect queue manager

```
CALL 'MQDISC' USING Hconn
       CompCode
       Reason
END-CALL.
```

The MQDISC call breaks the connection between the queue manager and the application program, and is the inverse of MQCONN.

When the application has finished all interaction with the queue manager, the application must sever the connection by means of the MQDISC call.

After the call, the connection handle (Hconn) is no longer valid, and message-queuing calls cannot be issued by the application unless it performs another MQCONN call.

Parameters

Hconn (MQHCONN) - input/output

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

On successful completion of the call, the queue manager sets this parameter to a value that is not a valid handle.

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_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_FAILED:

- MQRC_CONNECTION_BROKEN - Connection lost.
- MQRC_HCONN_ERROR - Connection handle not valid.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.

See “MQI return codes” on page 141, for more details

Guidelines

If an MQDISC call is issued when an application still has objects open, these objects are implicitly closed (with MQCO_NONE).

MQPUT1 - put one message

```
CALL 'MQPUT1' USING Hconn
        ObjDesc
        MsgDesc
        PutMsgOpts
        BufferLength
        Buffer
        CompCode
        Reason
END-CALL.
```

The MQPUT1 call puts one message on a queue; the queue need not be open.

For some applications, the typical sequence of calls to MQOPEN, multiple MQPUTS, and finally MQCLOSE is an efficient method for putting many messages onto a queue. For applications where only a single put is required, such as a remote database update for a single record, the MQPUT1 call can be used.

The MQPUT1 call is equivalent in function to the sequence of an MQOPEN call, followed by an MQPUT, and finally an MQCLOSE call, but only requires a single call.

The application specifies the handle for the queue manager (Hconn), the queue to put the information (ObjDesc), information about the message to be put (MsgDesc), options that control the action of the put (PutMsgOpts), the length of the data (BufferLength), and the message itself (Buffer).

Parameters

Hconn (MQHCONN) - input

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

ObjDesc (MQOD) - input

Object descriptor.

This is a structure which identifies the queue to which the message is added. See MQOD in “MQOD - MQ object descriptor structure” on page 132, for the format of the object descriptor.

The application must be authorized to open the queue for output.

MsgDesc (MQMD) - input/output

Message descriptor.

This structure describes the attributes of the message being sent, and receives feedback information after the put request is complete. See MQMD in “MQMD - MQ message descriptor structure” on page 133, for the format of the message descriptor.

PutMsgOpts (MQPMO) - input/output

Options that control the action of the MQPUT1 call.

See MQPMO in “MQPMO - MQ put message options structure” on page 138, for details.

BufferLength (MQLONG) - input

Length of the message in Buffer.

Zero is valid, and indicates that the message contains no application data.

Buffer (MQBYTExBufferLength) - input

This is a buffer containing the application data to be sent.

CompCode (MQLONG) - output

Completion Code.

It is one of the following:

- MQCC_OK - Successful completion.
- MQCC_WARNING - Warning (partial completion).
- MQCC_FAILED - Call failed.

Reason (MQLONG) - output

Reason code qualifying CompCode.

If CompCode is MQCC_OK:

- MQRC_NONE - No reason to report.

If CompCode is MQCC_WARNING:

- MQRC_PRIORTY_EXCEEDS_MAXIMUM - A priority > 0 was specified and is ignored by MQSeries System.

If CompCode is MQCC_FAILED:

- MQRC_ALIAS_BASE_Q_TYPE_ERROR - Alias base queue not a valid type.
- MQRC_BUFFER_LENGTH_ERROR - Buffer length parameter not valid.
- MQRC_CONNECTION_BROKEN - Connection lost.
- MQRC_EXPIRY_ERROR - Expiry time not valid.
- MQRC_FEEDBACK_ERROR - Feedback code not valid.
- MQRC_HANDLE_NOT_AVAILABLE - No more handles available.
- MQRC_HCONN_ERROR - Connection handle not valid.
- MQRC_MD_ERROR - Message descriptor not valid.
- MQRC_MISSING_REPLY_TO_Q - Missing reply-to-queue.
- MQRC_MSG_TOO_BIG_FOR_Q - Message length greater than maximum for queue.
- MQRC_MSG_TYPE_ERROR - Message type in message descriptor not valid.
- MQRC_OBJECT_TYPE_ERROR - Object type not valid.
- MQRC_OD_ERROR - Object descriptor structure not valid.
- MQRC_OPTIONS_ERROR - Options not valid or not consistent.
- MQRC_PERSISTENCE_ERROR - Persistence not valid.
- MQRC_PMO_ERROR - Put-message-options structure not valid.
- MQRC_PRIORITY_ERROR - Priority not valid.
- MQRC_PUT_INHIBITED - Puts inhibited for queue.
- MQRC_Q_FULL - Queue already at maximum depth.
- MQRC_Q_SPACE_NOT_AVAILABLE - No space available on disk for queue.
- MQRC_REPORT_OPTIONS_ERROR - Report options in message descriptor not valid.
- MQRC_STORAGE_NOT_AVAILABLE - Insufficient storage available.
- MQRC_UNEXPECTED_ERROR - Unexpected error occurred.
- MQRC_UNKNOWN_ALIAS_BASE_Q - Unknown alias base queue.
- MQRC_UNKNOWN_OBJECT_NAME - Unknown object name.
- MQRC_UNKNOWN_OBJECT_Q_MGR - Unknown object queue manager.
- MQRC_UNKNOWN_REMOTE_Q_MGR - Unknown remote queue manager.

See “MQI return codes” on page 141, for more details

Guidelines

1. The MQPUT1 call can be used when a single message is to be added to a queue. It is functionally equivalent to the MQOPEN, MQPUT, MQCLOSE sequence of calls.
2. If several messages are to be added to the same queue, it is advisable to open the queue explicitly using an MQOPEN, and then use repeated MQPUT calls before closing the queue using an MQCLOSE. This gives better performance than repeated use of the MQPUT1.

MQINQ - inquire about object attributes

```
CALL 'MQINQ' USING Hconn
        Hobj
        SelectorCount
        Selectors
        IntAttrCount
        IntAttrs
        CharAttrLength
        CharAttrs
        CompCode
        Reason
END-CALL.
```

The MQINQ call returns an array of integers and a set of character strings that contain the attributes of a specified queue.

Sometimes an application needs to determine one or more of the properties of a queue, in order to take appropriate action. For example, a load-balancing program might want to determine the current depth of the queue (that is the number of messages on the queue), so that the application could start another task if the number of queued messages has exceeded the capacity of the current number of tasks.

The attributes of the queue can be determined by means of the MQINQ call.

The application specifies the queue whose attributes are to be queried (Hobj), the number of attributes required (SelectorCount), and the selector codes for those attributes (Selectors). The application receives in return the values for those attributes (IntAttrs and CharAttrs).

In order to use the MQINQ call, the queue must first be opened for inquiry using the MQOPEN call.

Parameters

Hconn (MQHCONN) - input

Connection handle.

This handle represents the connection to the queue manager, and is returned by the MQCONN call.

Hobj (MQHOBJ) - input

Object handle.

This handle represents the object whose attributes are required. The handle must have been returned by an MQOPEN call with the MQOO_INQUIRE option.

SelectorCount (MQLONG) - input

Count of selectors.

This is the count of selectors that are supplied in the Selectors array. It is the number of attributes that are to be returned. Zero is a valid value. The maximum value allowed is 256.

Selectors (MQLONGxSelectorCount) - input

Array of attribute selectors.

This is an array of SelectorCount attribute selectors; each selector identifies an attribute (integer or character) whose value is required.

Each selector must be valid for the type of object that Hobj represents. If the object is a queue, and the selector is:

- Not a valid selector for queues of any type, an error is raised.
- Only applicable to queues of type, or types, other than that of the object, the call completes with a warning.

Selectors can be specified in any order. Attribute values that correspond to integer attribute selectors (MQIA_* selectors) are returned in IntAttrs in the same order in which these selectors occur.

Attribute values that correspond to character attribute selectors (MQCA_* selectors) are returned in CharAttrs in the same order in which those selectors occur. MQIA_* selectors can be interleaved with the MQCA_* selectors; only the relative order within each type is important.

If all the MQIA_* selectors occur first, the same element numbers can be used to address corresponding elements in the Selectors and IntAttrs arrays.

For each MQCA_* selector in the following descriptions, the constant that defines the length in bytes of the resulting string CharAttrs is given.

The following are valid for *any queue type*:

- MQIA_DEF_PERSISTENCE - Default persistence.
- MQIA_INHIBIT_PUT - Whether put operations are allowed.
- MQCA_Q_DESC - Queue description (MQ_Q_DESC_LENGTH).
- MQCA_Q_NAME - Queue name (MQ_Q_NAME_LENGTH).
- MQIA_Q_TYPE - Queue type.

The following are valid for *local queues*:

- MQCA_CREATION_DATE - Queue creation date (MQ_CREATION_DATE_LENGTH).
- MQCA_CREATION_TIME - Queue creation time (MQ_CREATION_TIME_LENGTH).
- MQIA_CURRENT_Q_DEPTH - Current queue depth.
- MQIA_DEFINITION_TYPE - Queue definition type.
- MQIA_INHIBIT_GET - Whether GET operations are allowed.
- MQCA_INITIATION_Q_NAME - Initiation queue name (MQ_Q_NAME_LENGTH)
- MQIA_MAX_MSG_LENGTH - Maximum message length.
- MQIA_MAX_Q_DEPTH - Maximum queue depth.
- MQIA_OPEN_INPUT_COUNT - Number of MQOPEN calls that have a queue open for input.
- MQIA_OPEN_OUTPUT_COUNT - Number of MQOPEN calls that have the queue open for output.
- MQCA_PROCESS_NAME - Name of process definition for queue (MQ_PROCESS_NAME_LENGTH).
- MQIA_SHAREABILITY - Whether queue can be shared.
- MQIA_TRIGGER_CONTROL - Trigger control.
- MQIA_TRIGGER_TYPE - Trigger type.
- MQIA_USAGE - Usage.

The following are valid for *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).

The following is valid for *alias queues*:

- MQCA_BASE_Q_NAME - Name of queue resolved to (MQ_Q_NAME_LENGTH).
- MQIA_INHIBIT_GET - Whether GET operation are allowed.

The following is valid for *transmission queues*:

- MQCA_XMIT_Q_NAME - Name of local transmission queue.

IntAttrCount (MQLONG) - input

Count of integer attributes.

This is the number of elements in the IntAttrs array. Zero is a valid value if there are no MQIA_* selectors in Selectors.

If this is at least the number of MQIA_* selectors in the Selectors parameter, all integer attributes requested are returned.

IntAttrs (MQLONGxIntAttrCount) - output

This is an array of IntAttrCount integer attribute values.

Integer attribute values are returned in the same order as the MQIA_* selectors in the Selectors parameter. If the array contains more elements than the number of MQIA_* selectors, the excess elements are unchanged.

If Hobj represents a queue, but an attribute selector is not applicable to that type of queue, the specific value MQIAV_NOT_APPLICABLE is returned for the corresponding element in the IntAttrs array.

If the IntAttrCount or SelectorCount parameter is zero, IntAttrs is not referenced; in this case, the parameter address passed by programs written in C may be null.

CharAttrLength (MQLONG) - input

Length of character-attributes buffer.

This is the length in bytes of the CharsAttrs parameter.

This must be at least the sum of the lengths required to hold each attribute string (see Selectors). Zero is a valid value if there are no MQCA_* selectors in Selectors.

CharAttrs (MQCHARxCharAttrLength) - output

Character attributes.

This is the buffer in which the character attributes are returned, concatenated together. The length of the buffer is given by the `CharAttrLength` parameter.

Character attributes are returned in the same order as the `MQCA_*` selectors in the `Selectors` parameter. The length of each attribute string is fixed for each attribute (see Selectors), and the value in it is padded to the right with blanks if necessary.

If the buffer is larger than is needed to contain all of the requested character attributes (including padding), the excess, beyond the last attribute returned, is unchanged.

If `Hobj` represents a queue, but an attribute selector is not applicable to that type of queue, a character string consisting entirely of asterisks (*) is returned as the value of that attribute in `CharAttr`.

If the `CharAttrLength` or `SelectorCount` parameter is zero, `CharAttrs` is not referenced; in this case, the parameter address passed by programs written in C may be null.

CompCode (MQLONG) - output

Completion code.

It is one of the following:

- `MQCC_OK` - Successful completion.
- `MQCC_WARNING` - Warning (partial completion).
- `MQCC_FAILED` - Call failed.

Reason (MQLONG) - output

Reason code qualifying CompCode.

If CompCode is `MQCC_OK`:

- `MQRC_NONE` - No reason to report.

If CompCode is `MQCC_WARNING`:

- `MQRC_SELECTOR_NOT_FOR_TYPE` - Selector not applicable for queue type.
- `MQRC_INT_ATTR_COUNT_TOO_SMALL` - Not enough space allowed for integer attributes.
- `MQRC_CHAR_ATTRS_TOO_SMALL` - Not enough space allowed for character attributes.

These reason codes are returned in the above order of preference if more than one applies.

If CompCode is `MQCC_FAILED`:

- `MQRC_CHAR_ATTR_LENGTH_ERROR` - Length of character attributes not valid.
- `*MQRC_CHAR_ATTRS_ERROR` - Character Attribute string not valid.
- `MQRC_CONNECTION_BROKEN` - Connection lost.
- `MQRC_HCONN_ERROR` - Connection handle not valid.
- `MQRC_HOBJ_ERROR` - Object handle not valid.
- `MQRC_INT_ATTR_COUNT_ERROR` - Count of integer attributes not valid.
- `*MQRC_INT_ATTRS_ARRAY_ERROR` - Integer attributes array not valid.
- `MQRC_NOT_OPEN_FOR_INQUIRE` - Queue object not open for inquire.
- `MQRC_SELECTOR_COUNT_ERROR` - Count of selectors not valid.
- `MQRC_SELECTOR_ERROR` - Attribute selector not valid.
- `MQRC_SELECTOR_LIMIT_EXCEEDED` - Count of selectors too big.
- `MRQC_STORAGE_NOT_AVAILABLE` - Storage not available
- `MQRC_UNEXPECTED_ERROR` - Unexpected error occurred.

See “MQI return codes” on page 141, for more details

Guidelines

1. The values returned are a snapshot of the selected attributes. There is no guarantee that the attributes will not change before the application can act upon the returned values.
2. See “MQSeries System configuration elements” on page 23, for more information about queue types and attributes.

MQI data types and structures

This section will examine the data types used by the MQI and will then present the primary data structures important to the MQI functions.

Data types

The following data types are used by the message queuing services in the MQSeries System:

- Elementary
- Structure

All user-defined data types ultimately resolve to elementary data types, or to aggregates of elementary types (arrays or structures).

Elementary data types

Message queuing uses the following elementary data types:

- MQBYTE - A single byte (string of eight bits)
- MQCHAR - A single character in a defined character set
- MQLONG - A four-byte signed binary integer

MQBYTE - Byte

The MQBYTE data type represents a single byte of data. No particular interpretation is placed on the byte. The byte is treated as a string of bits, and not as a character or binary number. No special alignment is required.

An array of MQBYTE is sometimes used to represent an area of main storage whose nature is not known to the queue manager. For example, the area may contain application message data or a structure. The boundary alignment of this area must be compatible with the nature of the data it contains.

MQBYTE24 - String of 24 Bytes

A string of 24 bytes. Each byte is described by the MQBYTE data type.

MQBYTE 32 - String of 32 Bytes

A string of 32 bytes. Each byte is described by the MQBYTE data type.

MQCHAR - Character

The MQCHAR data type represents a single character. The coded character set identifier of the character is that of the queue manager. No special alignment is required.

Note: Application message data specified on MQGET, MQPUT, and MQPUT1 calls is described by the MQBYTE data type.

MQCHARn - String of n Characters

Each MQCHARn data type represents a string of *n* characters, where *n* can take one of the following values:

4, 8, 12, 16, 28, 32, 48, 64, 128, 256

Each character is described by the MQCHAR data type. No special alignment is required.

If the data in the string is shorter than the defined length of the string, the data must be padded with blanks to fill the string. In some cases, a null character can be used to end the string prematurely, instead of padding with blanks.

Characters beyond the null character, up to the defined length of the string, are ignored. Cases where null characters may be used are identified in the call and data type descriptions.

When the queue manager returns character strings to the application (for example, on the MQGET call), the queue manager always pads with blanks to the defined length of the string.

Constants are available that define the lengths of the character string fields.

MQHCONN - Connection Handle

The MQCONN data type represents a *queue manager* connection handle. The MQCONN data type is defined as an MQLONG, and must be aligned on a 4-byte boundary.

Applications must only test variables of this type for equality.

MQHOBJ - Object Handle

The MQHOBJ data type represents an object, (*queue*) handle. The MQHOBJ is defined as an MQLONG, and must be aligned on a 4-byte boundary.

Applications must only test variables of this type for equality.

MQLONG - Long Integer

The MQLONG data type is a 32-bit signed binary integer that can take any value in the range -2147483648 through +2147483647, unless otherwise restricted by the context.

For COBOL, the valid range is limited to -999 999 999 through +999 999 999.

An MQLONG must be aligned on a 4-byte boundary.

Structure data types

The supported programming languages vary in their functionality with respect to structures, and certain rules and conventions are adopted in mapping the message-queuing structure data types of each programming language.

Boundary alignments

1. Structures are aligned on their natural boundaries. All message-queuing structures require 4-byte alignment.
2. Each field in the structure is aligned on its natural boundary. Fields of type MQLONG are aligned on 4-byte boundaries. Other fields are aligned on 1-byte boundaries.
3. The length of a structure is a multiple of its boundary requirement. All message-queuing structures have lengths that are multiples of four bytes.
4. Padding fields are declared explicitly where necessary to ensure compliance with rules 2 and 3.

References to structure components

The supported programming languages allow references to structure components to be qualified with the name of the structure. Multiple instances of the structure may be declared:

- COBOL has the IN keyword

Characters in names

The supported programming languages accept mixed case, however, the following points should be noted:

- The COBOL language is not case sensitive, and so the names may be coded in lowercase, mixed case, or all uppercase. However, the following changes must be made:
  - The underscore character (_) used in the names of constants must be replaced by the hyphen (-) character.
  - The names of structure fields must be prefixed with the name of the structure followed by a hyphen.

MQOD - MQ object descriptor structure

The `MQOD` structure is used to specify a queue object.

This structure is passed as a parameter to the `MQOPEN` and `MQPUT1` calls.

StrucId (`MQCHAR4`)

Structure identifier.

The value must be:

`MQOD_STRUC_ID`

Structure identifier for Object Descriptor.

This is always an input field.

Version (`MQLONG`)

Structure version number.

The value must be:

`MQOD_VERSION_1`

Structure version number for Object Descriptor.

This is always an input field.

ObjectType (`MQLONG`)

Object type.

Type of object being named in ObjectName. This must be:

`MQOT_Q`

Queue.

This is an input field.

ObjectName (`MQCHAR48`)

Object name.

The local name of the object as defined on the queue manager identified by `ObjectQMgrName`.

The name must not contain leading or embedded blanks, but may contain trailing blanks. The first null character and characters following it are treated as blanks.

This is an input field.

ObjectQMgrName (`MQCHAR48`)

Object queue manager name.

The name of the queue manager on which the `ObjectName` object is defined.

If the name is specified, it must not contain leading or embedded blanks, but may contain trailing blanks. The first null character and characters following it are treated as blanks.

A name which is entirely blank up to the first null character or the end of the field denotes the queue manager to which the application is connected.

This is an input field.

DynamicQName (MQCHAR48)

This is a reserved field.

AlternateUserId (MQCHAR12)

This is a reserved field.

MQMD - MQ message descriptor structure

The MQMD structure is used to describe the attributes of a message. It is an input/output variable for MQGET, MQPUT, and MQPUT1 calls.

StrucId (MQCHAR4)

Structure identifier.

The value must be:

MQMD_STRUC_ID

Structure identifier for Message Descriptor.

This is always an input field.

Version (MQLONG)

Structure version number.

The value must be:

MQMD_VERSION_1

Structure version number for Message Descriptor.

This is always an input field.

Report (MQLONG)

Reserved.

This is a reserved field. The value must be 0 (zero).

MsgType (MQLONG)

Message type.

This indicates the type of the message. It must be one of the following:

- MQMT_REQUEST - Message requiring reply.
- MQMT_REPLY - A reply to earlier request message.
- MQMT_DATAGRAM - A message not requiring a reply.
- MQMT_REPORT - A report message.

The description for these options follow.

MQMT_REQUEST

This message is one requiring a reply.

MQMT_REPLY

This message is the reply to an earlier request message (MQMT_REQUEST). The message should be sent to the queue indicated by the ReplyToQ field of the request message.

Note: The queue manager does not enforce the request-reply relationship. The request-reply relationship is the responsibility of the application.

MQMT_DATAGRAM

The message is one which does not require a reply.

MQMT_REPORT

The message is reporting on some unexpected occurrence (for example, a request message was received which contained data which was not valid).

The message should be sent to the queue indicated by ReplyToQ field of the message descriptor of the message which caused the error.

The Feedback field should be set to indicate the nature of the report. In addition, the CorrelId field of the report message should be set to the message identifier of the message which caused the error.

This is an output field for the MQGET call, and an input field for MQPUT and MQPUT1 calls.

Expiry (MQLONG)

Reserved.

This is a reserved field. The value must be -1.

Feedback (MQLONG)

Feedback code.

This is used with a message of type MQMT_REPORT to indicate the nature of the report, and is only meaningful with that type of message.

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 which generate report messages should not use feedback codes in the system range, other than MQFB_QUIT.

On MQPUT or MQPUT1 calls, the value specified must be within either the system range or the user range.

A special feedback code is:

MQFB_QUIT

Application should end. This can be used by a workload scheduling program to control the number of instances of an application program that are running. Sending an MQMT_REPORT message with this feedback code to an instance of the application program indicates to that instance that it should stop processing. However, adherence to this convention is a matter for the application. It is not enforced by the queue manager.

This is an output field for the MQGET call, and an input field for the MQPUT and MQPUT1 calls.

Encoding (MQLONG)

Data encoding.

This identifies the representation used for the numeric values in the application message data. This applies to binary integer data, packed-decimal integer data, and floating-point data.

The following value is defined:

MQENC_NATIVE

Native machine encoding.

The encoding is the same as that of the machine on which the application is running.

Note: The value of this constant is environment-specific.

Applications should normally specify the MQENC_NATIVE.

This is an output field for the MQGET call, and an input field for MQPUT and MQPUT1 calls.

CodedCharSetId (MQLONG)

Coded character-set identifier.

This specifies the coded character-set identifier of character data in the user message data.

Note that character data in the message descriptor and the other message queuing data structures must be in the character set used by the queue manager.

The following special value may be specified:

MQCCSI_Q_MGR

Queue manager's coded character-set identifier.

Character data in the user message data is in the queue manager's character set.

On MQPUT and MQPUT1 calls, the queue manager changes the value MQCCSI_Q_MGR to the value of the queue manager's CodedCharSetId attribute. MQCCSI_Q_MGR is never returned by the MQGET call.

This is an output field for the MQGET call, and an input field for MQPUT and MQPUT1 calls.

Format (MQCHAR8)

Format name.

This is the name that the sender of the message may use to indicate to the receiver the nature of the data in the message. Any characters that are in the queue manager's character set may be specified for the name, but it is recommended that the name be restricted to the following:

- Uppercase A through Z
- Numeric digits 0 through 9
- Blank
- Null Character

If other characters are used, it may not be possible to translate the name between the character sets of the sending and receiving queue managers.

If there is a null character, it and any subsequent characters are treated as blanks. For the MQGET call, the queue manager returns the name padded with blanks to the length of the field.

Do not use names beginning with "MQ". Names beginning with "MQ" are reserved for use by the queue manager.

This is an output field for the MQGET call, and an input field for the MQPUT and MQPUT1 calls.

Priority (MQLONG)

Message priority.

The MQSeries System ignores priority. However, positive values for this parameter may be specified and the MQSeries System will propagate the value along with the message to the destination. Use of values greater than zero in this field will generate an MQCC_WARNING.

Persistence (MQLONG)

Message persistence.

For MQPUT and MQPUT1 calls, the value must be one of the following:

- MQPER_PERSISTENT - Message is persistent. The message survives restarts of the queue manager. When a persistent message is sent to a remote queue, a store-and-forward mechanism is used to hold the message on a local queue manager instance until it is known to have arrived at the next destination (input and output values).
- MQPER_NOT_PERSISTENT - Message not persistent. The message does not survive restarts of the queue manager. This option is not supported by the MQSeries System.

For an MQGET call, the value returned is either MQPER_PERSISTENT or MQPER_NOT_PERSISTENT.

This is an output field for the MQGET call, and an input field for MQPUT and MQPUT1 calls (only MQPER_PERSISTENT can be set).

MsgId (MQBYTE24)[3]

Message identifier.

On return from an MQGET call, the MsgId field is set to the message identifier of the message returned (if any).

For MQPUT and MQPUT1 calls, if MQMI_NONE is specified by the application, the queue manager generates a unique message identifier that it places in the message descriptor sent with the message.

The queue manager also returns this message identifier in the message descriptor belonging to the sending application. The application can use this value to record information about particular messages, and to respond to queries from other parts of the application.

The sending application can also specify a particular value for the message identifier, other than MQMI_NONE. This stops the queue manager generating a unique message identifier. This facility can be used by an application that is forwarding a message, to propagate the message identifier of the original message.

The queue manager does not itself make any use of this field except to:

- Generate a unique value if requested.
- Deliver the value to the application that issued the get request for the message.

This field is not subject to any translation based on the character set of the queue manger. The field is treated as a string of bits.

The following special value may be used:

MQMI_NONE

No message identifier is specified. The value is binary zero for the length of the field. For the MQGET call, MQMI_NONE must be specified, and the first available message on the queue will be returned.

This is an input/output field for MQGET, MQPUT and MQPUT1 calls.

CorrelId (MQBYTE24)

Correlation identifier.

On return from an MQGET call, the CorrelId field is set to the correlation identifier of the message returned (if any).

For MQPUT and MQPUT1 calls, the application can specify any value. The queue manager transmits this value with the message and delivers it to the application that issued the get request for the message.

The field is not subject to any translation based on the character set of the queue manager. The field is treated as a string of bits.

The following special value may be used:

MQCI_NONE

No correlation identifier is specified.

The value is binary zero for the length of the field. For the MQGET call, MQCI_NONE must be specified, and the first available message on the queue will be returned.

This is an input/output field for MQGET calls, and an input field for MQPUT and MQPUT1 calls.

BackoutCount (MQLONG)

This is a reserved field.

3. A generated MsgId consists of a 4-byte product identifier followed by a product-specific implementation of a unique number. There is no guarantee that queue manager-generated MsgId values do not clash with application-generated ones.

ReplyToQ (MQCHAR48)[4]

Name of reply queue.

The name of the message queue to which the application that issued the get request for the message should send MQMT_REPLY and MQMT_REPORT messages. The name is the local name of a queue that is defined on the queue manager identified by ReplyToQMgr.

For MQPUT and MQPUT1 calls, this field is required if an MQMT_REQUEST type message is specified in the message descriptor. However, the value specified is passed on to the application that issued the get request for the message, whatever the message type.

No check is made at the time of the MQPUT and MQPUT1 call that this name is known to the queue manager, or satisfies the naming rules for queues. Otherwise the only check made is that the name is not null, if it is required.

If the name is specified, it should not contain leading or embedded blanks, but it may contain trailing blanks. The first null character and characters following the null, are treated as blanks. A name that is entirely blank up to the first null character or the end of the field indicates that there is no reply-to-queue.

For the MQGET call, the queue manager always returns the name padded with blanks to the length of the field.

The queue specified must be able to be opened for output by the application that receives the request message. The application design must ensure that the necessary queues exist and are appropriately authorized.

This is an output field for the MQGET call, and an input field for the MQPUT and MQPUT1 calls.

ReplyToQMgr (MQCHAR48)[5]

Name of the reply queue manager.

The name of the queue manager to which the reply message is sent. ReplyToQ is the local name of a queue that is defined to that queue manger.

No check is made at the time of the MQPUT or MQPUT1 call that this name is known to the queue manager, or satisfies the naming rules for queue managers.

If the name is specified, it should not contain leading or embedded blanks, but it may contain trailing blanks. The first null character and characters following it are treated as blanks. A name that is entirely blank up to the first null character or the end of the field denotes the queue manager to which the application is connected.

For an MQGET call, the queue manager always returns the name padded with blanks to the length of the field.

This is an output field for the MQGET call, and an input field for the MQPUT and calls.

UserIdentifier (MQCHAR12)

This is a reserved field.

AccountingToken (MQBYTE32)

This is a reserved field.

ApplIdentityData (MQCHAR36)

This is a reserved field.

PutApplType (MQLONG)

This is a reserved field.

PutApplName (MQCHAR28)

This is a reserved field.

4. These may be defined as Alias Reply Queue via Queue Definition screen in the Configuration Section.
5. These may be defined as Alias Reply Queue via Queue Definition screen in the Configuration Section.

PutDate (MQCHAR8)

This is a reserved field.

PutTime (MQCHAR8)

This is a reserved field.

ApplOriginData (MQCHAR4)

This is a reserved field.

MQPMO - MQ put message options structure

StrucId (MQCHAR4)

Structure identifier.

The value must be:

MQPMO_STRUC_ID

Structure identifier for Put-Message Options.

This is always an input field.

Version (MQLONG)

Structure version number.

The value must be:

MQPMO_VERSION_1

Structure version number for Put-Message Options.

This is always an input field.

Options (MQLONG)

Options.

There is only one option supported:

- MQPMO_SYNCPOINT - This option implies that this platform supports syncpointing.

  The description of this option follows.

  MQPMO_SYNCPOINT

  This option is implied because CICS/VSE only supports message syncpointing.

Timeout (MQLONG)

This is a reserved field.

Context (MQHOBJ)

This is a reserved field.

KnownDestCount (MQLONG)

This is a reserved field.

UnknownDestCount (MQLONG)

This is a reserved field.

InvalidDestCount (MQLONG)

This is a reserved field.

ResolvedQName (MQCHAR48)

Resolved name of the destination queue.

This is an output field that is set by the queue manager to the name of the queue that received the message after alias resolution. This can be either a local queue name or a remote queue name.

In each case the name is the local name of a queue that is defined on the queue manager identified by ResolvedQMgrName.

ResolvedQMgrName (MQCHAR48)

Resolved name of destination queue manager.

The name of the queue manager that received the message after alias resolution. ResolvedQName is the local name of a queue that is defined on that queue manager.

This is an output field

MQGMO - MQ get message options structure

The MQGMO structure is an input variable for passing the MQGET call.

StrucId (MQCHAR4)

Structure identifier.

The value must be:

MQGMO_STRUC_ID

Structure identifier for Get-Message Options.

This is always an input field.

Version (MQLONG)

Structure version number.

The value must be:

MQGMO_VERSION_1

Structure version number for Get-Message Options.

This is always an input field.

Options (MQLONG)

Options.

Any or none of the following can be specified. If more than one is required, the values are added together.[6] Combinations that are not valid are noted. All other combinations are valid. The following options are supported:

- MQGMO_WAIT - Wait for message to arrive.
- MQGMO_NO_WAIT - Return immediately if no suitable message.
- MQGMO_BROWSE_FIRST - Browse from start of queue.
- MQGMO_BROWSE_NEXT - Browse from current position.
- MQGMO_ACCEPT_TRUNCATED_MSG - Allow truncation of message data.
- MQGMO_MSG_UNDER_CURSOR - Get message under browse cursor.
- MQGMO_SYNCPOINT - This option implies that this platform supports syncpointing.
- MQGMO_LOCK - Perform Browse message lock on MQGET.
- MQGMO_UNLOCK - Unlock prior lock record.

The description of these options follows.

MQGMO_WAIT

The application is to wait until a message arrives. The maximum time the application waits is specified in WaitInterval.

If get requests are inhibited, this call returns with an error, whether or not there are any messages on the queue. If get requests become inhibited while this call is waiting, it returns immediately with an error.

This option can be used with the MQGMO_BROWSE_FIRST or MQGMO_BROWSE_NEXT options.

If several applications are waiting on the same shared queue, all the applications are activated when a suitable message arrives.

MQGMO_NO_WAIT

6. Do not add the same constant more than once.

The application is not to wait if no suitable message is available. This is the opposite of the MQGMO_WAIT option, and is defined to aid program documentation. It is the default if neither is specified.

MQGMO_BROWSE_FIRST

The MQGET call is the first in a browse sequence. This can be used in the middle of a browse sequence to reset the browse cursor to the start of the queue.

The first message in the queue, satisfying any conditions specified in the message descriptor, is returned to the application, but the message remains in the queue.

If the message is removed from the queue before the next MQGET call with the MQGMO_BROWSE_NEXT is issued, the browse cursor logically remains at the position in the queue that the message occupied, even though that position is now empty.

If the MQGET call is issued immediately after the MQOPEN call, this option has the same effect as MQGMO_BROWSE_NEXT.

After this call, the browse cursor is positioned logical on the message that has been returned.

The MQGMO_MSG_UNDER_CURSOR option can subsequently be used with a non-browse MQGET call is required, to remove the message from the queue.

Note that the browse cursor is not moved by non-browse MQGET calls using the same Hobj handle.

MQGMO_BROWSE_NEXT

The browse cursor is advanced to the next message on the queue, that satisfies any conditions specified in the message descriptor. The message is returned to the application, but remains on the queue.

If the message is removed from the queue before the next MQGET call with MQGMO_BROWSE_NEXT is issued, the browse cursor logically remains at the position in the queue that the message occupied, even though that position is now empty.

The MQGMO_MSG_UNDER_CURSOR option can subsequently be used with a non-browse MQGET call if required, to remove the message from the queue.

Note that the browse cursor is not moved by non-browse MQGET calls using the same Hobj handle.

MQGMO_ACCEPT_TRUNCATED_MSG

The MQGET operation completes successfully, with a warning. The message is removed from the queue if on a non-Browse request specifying MQGMO_MSG_UNDER_CURSOR(at the syncpoint, if applicable), even though the BufferLength is shorter than the message. Without this option, a buffer which is too small causes the MQGET to complete unsuccessfully.

MQGMO_SET_SIGNAL

Not Used.

MQGMO_MSG_UNDER_CURSOR

This option causes the message pointed to by the browse cursor to be retrieved, regardless of the values specified in the MsgId and CorrelId fields in the MsgDesc parameter.

The message pointed to by the browse cursor is the one that was last retrieved using either the MQGMO_BROWSE_FIRST or the MQGMO_BROWSE_NEXT option.

Note: This option must not be specified with either the MQGMO_BROWSE_FIRST or the MQGMO_BROWSE_NEXT option. It is also an error if the queue was not opened both for browse for input. If the browse cursor is not currently pointing to a retrievable message, an error is returned by the MQGET call.

MQGMO_SYNCPOINT

This option is implied because CICS/VSE only supports message syncpointing.

MQGMO_LOCK

This option can be used with Browse FIRST or NEXT and will cause a lock to be placed on the record given back to the application. This record will be under exclusive control by the application. This lock stays in effect until one of the following happens:

(1). An EXEC CICS syncpoint
(2). The application task ends
(3). A MQGET with just a MQGMO_UNLOCK option

MQGMO_UNLOCK

This option will unlock a prior MQGMO_LOCK request that was successful.

WaitInterval (MQLONG)

Wait interval.

The maximum time, expressed in milliseconds, that the MQGET call waits for a message to arrive. After this time, the call completes with an error (MQRC_NO_MSG_AVAILABLE).

MQWI_UNLIMITED specifies an infinite wait and in the absence of any message will terminate only at system shutdown.

This field is used in conjunction with the MQGMO_WAIT option. It is ignored if this option is not specified.

Signal1 (PMQLONG)

This is a reserved field its value is not significant.

Signal2 (MQLONG)

This is a reserved field its value is not significant.

ResolvedQName (MQCHAR48)

Resolved name of the destination queue.

This is an output field which is set by the queue manager to the local name of the queue from which the message was retrieved, as defined to the connected queue manager.

The resolved name is different from the name used to open the queue if an alias name was used. For the case of an alias queue, the name of the local queue is returned.

MQI return codes

For each MQI call, a completion code and a reason code are returned by the MQSeries System to indicate the success or failure of the MQI function. This section lists the possible codes.

MQI completion codes

The completion code CompCode) parameter informs the application making an MQI call whether or not the call completed successfully, completed partially, or failed.

The possible completion codes are as follows:

0 MQCC_OK

Successful completion.

The call completed fully. All output parameters have been set.

The Reason parameter always has the value MQRC_NONE in this case.

1 MQCC_WARNING

Warning of partially completed call.

The call completed partially. Some output parameters may have been set in addition to the CompCode and Reason output parameters.

The Reason parameter gives additional information.

2 MQCC_FAILED

Call failed.

The processing of the call did not complete. The state of the queue manager is normally unchanged (exceptions are specifically noted). Only the CompCode and Reason output parameters have been set.

The reason may be a fault in the application program, or the reason may be a result of some situation outside the application, for example the application's authority may have been revoked.

The Reason parameter gives additional information.

MQI reason codes

The reason code (Reason) parameter is a qualification to the CompCode.

If there is no special reason to report, MQRC_NONE is returned. A successful call typically returns MQCC_OK and MQRC_NONE.

If the CompCode is either MQCC_WARNING or MQCC_FAILED, the queue manager always reports a qualifying reason. Details are provided under each call description.

An alphabetical listing of all reason codes and descriptions follows.

Note: Reason codes marked with an asterisk (*) are not currently implemented.

0 MQRC_NONE

No reason to report.

The call completed normally (CompCode is MQCC_OK).

Corrective action: None.

2000 *MQRC_ACCESS_RESTRICTED

Queue manager in restricted access mode.

The MQCONN call was rejected because the queue manager has been started in restricted access mode.

Corrective action: Contact your system administrator.

2001 MQRC_ALIAS_BASE_Q_TYPE_ERROR

Alias base queue not a valid type.

An MQOPEN or MQPUT1 request was issued, specifying an alias queue as the target, but the BaseQName in the alias queue attributed resolves to a queue that is not predefined local or remote queue.

Corrective action: Correct the queue definitions.

2002 MQRC_ALREADY_CONNECTED

Application already connected.

An MQCONN call was issued, but the application is already connected to the queue manager.

Corrective action: None. The Hconn parameter returned has the same value as was returned for the previous MQCONN call.

2004 *MQRC_BUFFER_ERROR

Buffer parameter not valid.

Buffer is not valid. The parameter pointer is not valid, or points to read-only storage for MQGET calls, or to storage that cannot be accessed for the entire length specified by BufferLength. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Correct the parameter.

2005 MQRC_BUFFER_LENGTH_ERROR

Buffer length parameter not valid.

BufferLength is not valid. The reason may also be returned if the parameter pointer is not valid. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Specify a nonnegative value.

2006 MQRC_CHAR_ATTR_LENGTH_ERROR

Length of character attributes not valid.

CharAttrLength is negative (for MQINQ calls) or is not large enough to hold all selected attributes. This reason also occurs if the parameter pointer is not valid. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

2007 *MQRC_CHAR_ATTRS_ERROR

Character attributes string not valid.

CharAttrs is not valid. The parameter pointer is not valid, or points to read-only storage from MQINQ calls or to storage that is not as long as implied by CharAttrLength. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Correct the parameter.

2008 MQRC_CHAR_ATTRS_TOO_SHORT

Not enough space allowed for character attributes.

For MQINQ calls, CharAttrLength is not large enough to contain all of the character attributes for which MQCA_* selectors are specified in the Selectors parameter.

The call still completes, with the CharAttrs parameter string is filled in with as many character attributes as there is room for. Only complete attribute strings are

returned. Space at the end of the string that is not large enough to hold the next attribute is unchanged.

Corrective action: Specify a large enough value, unless only a subset of the values is needed.

2009 MQRC_CONNECTION_BROKEN

Connection not established.

Connection to the queue manager has been lost or was not established. This can occur because the MQCONN call was not executed.

Corrective action: Applications must establish connection by issuing the MQCONN call.

2010 *MQRC_DATA_LENGTH_ERROR

Data length parameter not valid.

DataLength is not valid. The parameter pointer is not valid, or points to read-only from storage. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Correct the parameter.

2013 MQRC_EXPIRY_ERROR

Expiry time not valid.

The Expiry field is reserved, and must have a value of -1.

Corrective action: Specify -1.

2014 MQRC_FEEDBACK_ERROR

Feedback code not valid.

A feedback code (Feedback) was specified in MQMD that is outside both the range defined for system feedback codes and that defined for application feedback codes.

Corrective action: Specify a valid value.

2016 MQRC_GET_INHIBITED

Gets failed for the queue.

MQGET calls have failed for the queue because the InhibitGet attribute has been set for the queue.

Corrective action: Clear the InhibitGet attribute via the administration screens.

2017 MQRC_HANDLE_NOT_AVAILABLE

No more handles available.

An MQOPEN or MQPUT1 request was issued, but the maximum number of open handles allowed has already been reached.

Corrective action: Check whether the application is looping. Otherwise, reduce the complexity of the application. Check the maximum number of open handles that the system can have in the queue-manager attribute (Maximum Open Queues).

2018 MQRC_HCONN_ERROR

Connection handle not valid.

Hconn is not valid. This reason occurs if the parameter pointer is not valid, or points to read-only storage for the MQCONN call. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Ensure that a successful MQCONN call is performed for the queue manager instance, and that an MQDISC call has not already been performed for it. Check that the handle is being used within its valid scope. See “MQCONN - connect queue manager” on page 112.

2019 MQRC_HOBJ_ERROR

Object handle not valid.

Hobj is not valid. This reason also occurs if the supplied value is incorrect, the parameter pointer is not valid, or points to a read-only storage for an MQOPEN call. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Ensure that a successful MQOPEN call is performed for this object, and that an MQCLOSE call has not already been performed for it. For MQGET and MQPUT calls, also ensure that the handle represents a queue object. Check that the handle is being used within its valid scope. See “MQOPEN - open message queue” on page 114.

2021 MQRC_INT_ATTR_COUNT_ERROR

Count of integer attributes not valid.

IntAttrCount is negative (for MQINQ calls), or is not large enough to hold all selected attributes. This reason also occurs if the parameter pointer is not valid. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Specify a value large enough for all selected integer attributes.

2022 MQRC_INT_ATTR_COUNT_TOO_SMALL

Not enough space allowed for integer attributes.

For MQINQ calls, IntAttrsCount is not as large as the number of integer attribute selectors (MQIA_*) specified in the Selectors parameter.

The call still completes, with the IntAttrs array filled with as many integer attributes as there is room for.

Corrective action: Specify a large enough value, unless only a subset of the values is needed.

2023 *MQRC_INT_ATTRS_ARRAY_ERROR

Integer attributes array not valid.

IntAttrs is not valid. The parameter pointer is not valid, or points to read-only storage for an MQINQ call or to storage that is not as long as indicated by IntAttrCount. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Correct the parameter.

2025 MQRC_MAX_CONNS_LIMIT_REACHED

Connection initialization failure.

The MQCONN call was rejected because the maximum number of connects has already been reached.

Corrective action: Check to make sure that the program is not looping or check the Queue Manager Attribute (Maximum number of MQCONN).

2026 MQRC_MD_ERROR

Message descriptor not valid.

MQMD control block is not valid. Either the StrucId mnemonic eye-catcher is not valid, or the Version is not recognized. This reason also occurs if the parameter pointer is not valid, or points to read only storage. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Correct the definition of the message descriptor. Ensure that the required input fields are correctly set.

2027 MQRC_MISSING_REPLY_TO_Q

Missing reply-to-queue.

The reply-to-queue name (ReplyToQ) in MQMD is not specified (that is, it is all blanks), but a reply was requested (MQMT_REQUEST was specified in the MsgType field of the message descriptor).

Corrective action: Specify the name of the queue to which the reply is to be sent.

2029 MQRC_MSG_TYPE_ERROR

Message type in message descriptor not valid.

Message type (MsgType) in the message descriptor (MQMD) is not valid.

Corrective action: Ensure that a valid type is specified.

2030 MQRC_MSG_TOO_BIG_FOR_Q

Message length greater than maximum for queue.

An attempt was made to put a message that is bigger than allowed by the queue.

Corrective action: Check whether BufferLength was correctly specified. If so, either break the message into several smaller messages, or increase MaxMsgLength for the queue.

2033 MQRC_NO_MSG_AVAILABLE

No message available.

An MQGET call was issued, on the queue, but there is no message that satisfies the request. Either the MQGMO_WAIT option was not specified or it was specified but the timeout interval has expired with no message arriving on the queue.

This message is also returned for an MQGET call for browse, if the browse sequence has been reached.

Corrective action: If this is an unexpected condition, check whether the message was successfully put on the queue.

Consider waiting longer for the message.

2034 MQRC_NO_MSG_UNDER_CURSOR

No message under cursor.

An MQGET was performed with the option MQGMO_MSG_UNDER_CURSOR but had not been preceded by a browse operation.

Corrective action: Establish the browse cursor by issuing MQGET with option MQGMO_BROWSE_* prior to issuing the MQGET which failed.

2035 *MQRC_NOT_AUTHORIZED

Not authorized for access.

On the MQCONN call, the application is not authorized to connect to the queue manager. On MQOPEN or MQPUT1 calls, the application is not authorized to open the object for the option, or options, specified.

Corrective action: Ensure that the correct queue manager or object was specified, and that appropriate authority exists.

2036 MQRC_NOT_OPEN_FOR_BROWSE

Queue object not open for browse.

An MQGET call was issued to a queue not opened for browse with one of the following options:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_MSG_UNDER_CURSOR

Corrective action: Specify MQOO_BROWSE when the queue is opened.

2037 MQRC_NOT_OPEN_FOR_INPUT

Queue object not open for input.

Corrective action: Specify MQOO_INPUT_EXCLUSIVE or MQOO_INPUT_SHARED when the queue is opened.

2038 MQRC_NOT_OPEN_FOR_INQUIRE

Queue object not open for inquire.

Corrective action: Specify MQOO_INQUIRE when the queue is opened.

2039 MQRC_NOT_OPEN_FOR_OUTPUT

Queue object not open for output.

Corrective action: Specify MQOO_OUTPUT when the queue is opened.

2041 *MQRC_OBJECT_CHANGED

Object definition changed since opened.

Since the Hobj handle used in this call was opened, object definitions that affect this object have been changed. See the MQOPEN call, in this chapter, for more information.

Corrective action: Issue an MQCLOSE call to return the handle to the system. Reopen the object, obtaining a new handle, and retry the operation.

If object definitions are critical to the application logic, an MQINQ call can be used to find out what has changed. See the MQINQ call, in this chapter, for more information.

2042 MQRC_OBJECT_IN_USE

Object already open with conflicting options.

An MQOPEN call has been issued, but the object in question has already been opened (by this or an other application), with options that conflict with those specified in the Options parameter. This arises if the request is for shared input, but the object is already open for exclusive input, and also if the request is for exclusive input, but the object is already open for input.

Corrective action: System design should specify whether an application is to wait and retry, or take other action.

2043 MQRC_OBJECT_TYPE_ERROR

Object type not valid.

ObjectType (in MQOD) is not valid because the field specifies an unrecognized value. The object type must be MQOT_Q.

Corrective action: Specify a valid object type.

2044 MQRC_OD_ERROR

Object descriptor structure not valid.

MQOD control block is not valid. Either the StrucId mnemonic eye-catcher is not valid, or the Version is not recognized. This reason also occurs if the parameter pointer is not valid, or points to read-only storage for an MQOPEN call for a dynamic queue. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results may occur.)

Corrective action: Correct the definition of the object descriptor. Ensure that required input fields are correctly set.

2045 MQRC_OPTION_NOT_VALID_FOR_TYPE

Option not valid for object type.

Option not valid for the type of queue being opened or closed, for example, MQOO_INQUIRE to a MQQT_REMOTE queue.

Corrective action: Specify the correct option.

2046 MQRC_OPTIONS_ERROR

Options not valid nor consistent.

The Options field or parameter is unrecognized, or contains a combination that is not valid.

For MQGET, MQPUT, or MQPUT1 calls, this field is in the options structure (MQGMO or MQPMO) for the call.

This reason also occurs if the Options parameter pointer is not valid for MQOPEN or MQCLOSE calls. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results may occur.)

Corrective action: Specify valid options. Check under the description of Options for the particular call, to see which option combinations are not valid.

2047 MQRC_PERSISTENCE_ERROR

Persistence not valid.

Persistence value in the message descriptor (MQMD) is not valid.

Corrective action: Specify a valid value.

2049 MQRC_PRIORITY_EXCEEDS_MAXIMUM

Put-message operation has specified a priority greater than zero.

This reason code is returned as a warning since the MQSeries System does not support message priorities. The priority value may be recognized by other MQSeries platforms.

Corrective action: As this is a warning, no corrective action is required. But, ensure the priority field is being properly initialized.

2050 MQRC_PRIORITY_ERROR[7]

Priority was negative value.

The Priority value in the message descriptor (MQMD) field has no effect, and must be specified as 0 or greater.

Corrective action: Specify a value greater than or equal to 0.

2051 MQRC_PUT_INHIBITED

Puts inhibited for the queue.

MQPUT and MQPUT1 calls are currently inhibited for the queue (InhibitPut), or for the queue to which the alias queue resolves.

Corrective action: If the system design allows applications to inhibit put requests for short periods, retry the operation later.

7. Any value greater than zero will call the MQRC_PRIORITY_EXCEEDS_MAXIMUM warning.

2053 MQRC_Q_FULL

Queue already at maximum depth.

The `MaxQDepth` limit setting has been reached.

Corrective action: Retry the operation later. Consider increasing the maximum depth for the queue, or arranging for additional instances of the application servicing the queue.

2056 MQRC_Q_SPACE_NOT_AVAILABLE

No space available on disk for queue.

An `MQPUT` or `MQPUT1` request was issued, but the request failed.

Corrective action: Review the error log for additional information.

2058 MQRC_Q_MGR_NAME_ERROR

Queue manager name not valid or not known.

The queue manager name specified for the `MQCONN` call is not valid. This reason also occurs if the parameter pointer is not valid. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results may occur.)

Corrective action: Use an all-blank name if possible, or verify the name used is valid.

2059 MQRC_Q_MGR_NOT_AVAILABLE

Queue manager initialization failed.

Corrective action: Review the error log for additional information.

2061 MQRC_REPORT_OPTIONS_ERROR

Report options in message descriptor not valid.

The `Report` field in the message descriptor (`MQMD`) is not valid.

Corrective action: Set the field to 0 (zero).

2063 *MQRC_SECURITY_ERROR

Security error occurred.

The `MQCONN` call was rejected because a security error occurred.

Corrective action: Note the error from the security manager, and contact your system programmer.

2065 MQRC_SELECTOR_COUNT_ERROR

Count of selectors not valid.

The `SelectorCount` parameter specifies a value which is not valid. This reason also occurs if the parameter pointer is not valid. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results may occur.)

Corrective action: Specify a value in the range 0 to 256.

2066 MQRC_SELECTOR_LIMIT_EXCEEDED

Count of selectors too big.

The SelectorCount parameter specifies a value larger than the maximum supported (256).

Corrective action: Reduce the number of selectors specified on the call. The valid range is 0 through 256.

2067 MQRC_SELECTOR_ERROR

Attribute selector not valid.

A selector in the Selectors array is not valid. This reason occurs if the parameter pointer is not valid. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results may occur.)

Corrective action: Ensure that the value specified for the selector is valid for the object type represented by Hobj.

2068 MQRC_SELECTOR_NOT_FOR_TYPE

Selector not applicable for queue type.

On the MQINQ call a selector in the Selectors array is not applicable to the type of queue whose attributes are being queried.

The call still completes, with the corresponding element, or elements, of IntAttrs set to MQIAV_NOT_APPLICABLE for an integer attribute, or the appropriate portion, or portions, of the CharAttrs string set to a character string of all asterisks (*).

Corrective action: Check the value specified in the selector.

2069 *MQRC_SIGNAL_OUTSTANDING

Signal outstanding for this handle.

An MQGET request was issued, with either the MQGMO_SET_SIGNAL or MQGMO_WAIT option, but there is already a signal outstanding for this object handle Hobj.

Corrective action: Check the application logic. If it is necessary to set a signal or wait when there is a signal outstanding for the same queue, a different object handle must be used.

2070 *MQRC_SIGNAL_REQUEST_ACCEPTED

No message returned, but signal request was accepted.

An MQGET request was issued, specifying MQGMO_SET_SIGNAL in the GetMsgOpts parameter. No suitable message is currently available. The application can now wait on the Signal1 field.

Corrective action: Wait on the Signal1 field and when the signal is delivered, check this field to ensure that a message is now available. If it is, reissue the MQGET request.

2071 MQRC_STORAGE_NOT_AVAILABLE

Internal error.

Corrective action: Review the error log for additional information.

2079 MQRC_TRUNCATED_MSG_ACCEPTED

Truncated message returned (message deleted from queue).

On an MQGET call, the message length was too large to fit in the supplied buffer. MQGMO_ACCEPT_TRUNCATED_MSG was specified, so the call completes. The message is removed from the queue (subject to syncpoint considerations), or, if this was a browse operation, the browse cursor advanced to this message.

The DataLength field is set by the system, and Buffer contains as much of the message as fits.

Corrective action: None, because the application expected this situation.

2080 MQRC_TRUNCATED_MSG_FAILED

Truncated message returned (message not deleted from queue).

On an MQGET call, the message length was too large to fit in the supplied buffer. MQGMO_ACCEPT_TRUNCATED_MSG was not specified, so the call fails. The message is not removed from the queue. If this was a browse operation, the browse cursor remains where it was before this call.

The DataLength field is set by the system, and Buffer contains as much of the message as fits.

Corrective action: Supply a large enough buffer, or specify MGMO_ACCEPT_TRUNCATED_MSG if not all of the message data is required.

2082 MQRC_UNKNOWN_ALIAS_BASE_Q

Unknown alias base queue.

An MQOPEN or MQPUT1 request was issued, specifying an alias queue as the target, but the BaseQName in the alias queue attributes is not recognized as a queue name.

Corrective action: Correct the queue definitions.

2085 MQRC_UNKNOWN_OBJECT_NAME

Unknown object name.

The ObjectName in the object descriptor (MQOD) is not recognized for the specified object type.

Corrective action: Specify a valid object name. Ensure that the name is padded to the right with blanks if necessary.

2086 MQRC_UNKNOWN_OBJECT_Q_MGR

Unknown object queue manager.

The ObjectQMgrName in the object descriptor (MQOD) is not valid for MQOPEN or MQPUT1.

Corrective action: Specify a valid queue manager name (or all blanks or an initial null character to refer to the connected queue manager instance).

2087 MQRC_UNKNOWN_REMOTE_Q_MGR

Unknown remote queue manager

An MQOPEN or MQPUT1 request was issued, specifying a remote queue as the target, but no suitable transmission queue has been defined.

Corrective action: Correct the queue definitions

2090 MQRC_WAIT_INTERVAL_ERROR

Negative wait interval in MQGMO.

A negative time-out (WaitInterval) value was specified in MQGMO (other than the special value MQWI_UNLIMITED).

Corrective action: Specify a value greater than or equal to zero, or MQWI_UNLIMITED.

2091 *MQRC_XMIT_Q_TYPE_ERROR

Transmission queue not local.

On an MQOPEN or MQPUT1 call, a message is to be sent to a remote queue manager and remote queue or alias queue manager. There is a queue defined on the connected queue manager with the same name as the remote queue manager, but this is not a local queue.

Corrective action: If a nonblank ObjectQMgrName was specified in the ObjDesc parameter, ensure that is was correct. Otherwise, correct the queue definitions.

2092 *MQRC_XMIT_Q_USAGE_ERROR

Transmission queue with wrong usage.

On an MQOPEN or MQPUT1 call, a message is to be sent to a remote queue manager and remote queue or alias queue manager. There is a local queue defined on the connected queue manager with the same name as the remote queue manager, but the local queue does not have a Usage of MQUS_TRANSMISSION.

Corrective action: Correct the queue definition.

2173 MQRC_PMO_ERROR

Put-message options structure not valid.

On an MQPUT or MQPUT1 call, the MQPMO structure is not valid. Either the StrucId mnemonic eye-catcher is not valid or the Version is not recognized. This reason also occurs if the parameter pointer is not valid, or points to read-only storage. (It is not always possible to detect an invalid parameter pointer, and if one is not detected, unpredictable results occur.)

Corrective action: Correct the definition of the MQPMO structure. Ensure that required input fields are correctly set.

2186 MQRC_GMO_ERROR

Get-message options structure not valid.

On an MQGET call, the MQGMO structure is not valid. Either the StrucId mnemonic eye-catcher is not valid, or the Version is not recognized. This reason also occurs if the parameter pointer is not valid, or points to read-only storage. (It is not always possible to detect parameter pointers which are not valid; if it is not detected, unpredictable results occur.)

Corrective action: Correct the definition of the MQGMO structure. Ensure that required input fields are correctly set.

2195 MQRC_UNEXPECTED_ERROR

Unexpected error occurred.

An error related to internal MQSeries System data structures has occurred. This is most likely due to trashing of memory due to an external hardware or software problem.

Corrective action: Contact your system administrator.

2206 MQRC_MSG_ID_ERROR

A MQGET call was issued with MQMD_MSGID having a search value other than LOW-VALUES. This is not supported.

Corrective action: MQMD_MSGID value should be cleared to LOW-VALUES before each MQGET.

2207 MQRC_CORREL_ID_ERROR

A MQGET call was issued with MQMD_CORRELID having a search value other than LOW-VALUES. This is not supported.

Corrective action: Move LOW-VALUES to MQMD_CORRELID field before each MQGET call.

2208 MQRC_FILE_SYSTEM_ERROR

An internal file error has occurred.

An internal file error has occurred while the Queue Manager was performing a request.

Corrective action: Get the error results from the error log, if present, and review the problem.

2209 MQRC_NO_MSG_LOCKED

A MQGET was issued with the MQGMO_UNLOCK option while no lock was being held for that Hobj.

Corrective action: Review application logic for possible prior error that caused a lock to not be held (i.e.: no more messages).

2210 *MQRC_LOCK_NOT_AVAILABLE

An internal MQI error has occurred.

This error is implementation specific. Examine the error log for additional information. An MQOPEN, MQGET, MQPUT, or MQPUT1 request was issued, and it was necessary to acquire a lock, but an internal error occurred.

Corrective action: Get the error code from the error log, if present, and review the problem.

Appendix A. System messages

The MQSeries system generates both internal and external messages. Internal messages are generated when an application program has activated the MQSeries system and an abnormal condition has occurred. These messages are stored on the System Log queue when it is available, otherwise, the CICS CSMT Transient Data Queue (TD) is used instead.

API system messages

The message structure of these messages comprise five 78-character lines of text.

Line 1 - "MQInnnnnn PRG:pppppppp TRN:tttt TRM:rrrr TSK:ccccc mm/dd/yy hh:mm:ss"

Where:

nnnnnn	MQSeries System message code
pppppppp	CICS Program name[1]
tttt	CICS Transaction code
rrrr	CICS Term id
ccccc	CICS Task id
mm/...:ss	Date and 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:rrrrrrrrrrrr EXEC LINE: lllll"

Where:

fff	EIBFN value at time of condition
rrrrrrrr	EIBRCODE
lllll	The DEBUG CICS Command Number

Line 7 - "EIBRESP: rrrrrrrr EIBRESP2: ssssssss EIBRSRCE:cccccccc ABCODE: aaaa"

Where:

rrrrrrrr	EIBRESP
ssssssss	EIBRESP2
cccccccc	EIBRSRCE
aaaa	CICS ABENDCODE

1. For CICS Program name and CICS Transaction code, see "Programs and transactions" on page 186.

MQSeries System message definitions

Note: In the following explanations of the messages, the message code and the textual description of the message, which normally appear on separate lines in the message, are shown on the same line for ease of use.

Each MQSeries System message listed below, alphabetically by message code, provides the following information:

- **Explanation**: This section tells 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 the user should be aware. The user may need to take further action.
8	An error message. An error has been detected and typically self correcting by the system but may require operator intervention.
10	A severe error message. An error has been detected which may severely affect user or system operation and requires immediate operator intervention.
12	A fatal error message. A error has been detected that is so severe that it causes one of the system components to terminate and requires immediate operator intervention.

- **Operator action**: If an operator response is necessary, this section tells what the appropriate responses are, and what their effect is. If this information is not shown, no operator response is required.
- **System action**: This part tells what is happening as a result of the condition causing the message or code. If this information is not shown, no system action is taken.

Message code (nnn):

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 terminated.

000004 SYNCH MSG DUP

Explanation : The received message may be duplicated.

Function : Receiver

Severity : 0

Operator Action : none

System Action : Continue on 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 the EIBFREE and the EIBERR fields are both not equal to low values.

For Program MQPSEND - As a Server upon completion of a RECEIVE command at least one of EIBERR, EIBRECV and EIBFREE does not equal to low values. As a Server or Sender upon receipt of an acknowledgment of messages sent the EIBFREE is not equal to low values and the EIBERR is equal to low values.

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 problem and restart communication.

System Action : Fatal error - Communication is terminated.

000011 LU62 EIB ERROR

Explanation : 1. As a Server upon completion of a RECEIVE the EIBERR not equal to low values.
2. As a Server or Sender upon receipt of an acknowledgment of messages sent, the EIBERR is not equal to low values.

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 problem and restart communication.

System Action : Fatal error - Communication is terminated.

000012 LU62 STAT ERROR

Explanation : As a Server or Sender upon receipt of an acknowledgment of messages sent, the EIBRECV is not equal to low values.

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 problem and restart communication.

System Action : Fatal error - Communication is terminated.

000013 LU62 ALLOC ERROR

Explanation : As a Sender upon completion of an ALLOCATE command, EIBRCODE is not equal to low values 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 terminated.

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 : 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.

System Action : Non-Fatal 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 problem and restart communication.

System Action : Fatal error - Communication is terminated.

000016 LU62 SEND ERROR

Explanation : As a Sender or Server upon completion of a SEND command, EIBRCODE is not equal to low values.

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 problem and restart communication.

System Action : Fatal error - Communication is terminated.

000017 REMOTE SITE DEALLOCATED CONVERSATION

Explanation : As a Server, if a get request returns a no message available response, a RECEIVE command is executed. Any of the following conditions will cause this response
1. EIBRECV is equal low values or
2. EIBFREE is equal low values or
3. EIBERR is equal low values.

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 terminate communication.

System Action : Communication is terminated.

000023 INVLD RESP TYPE

Explanation : The SENDER received a response message that doesn't conform to expected format.

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 receiver/requester process for programming error.

System Action : Fatal error - Communication is terminated.

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 sender process for programming error.

System Action : Fatal error - Communication is terminated.

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 sender process for programming error.
System Action :	Fatal error - Communication is terminated.

000051 QUEUE CONNECTION ERROR

Explanation :	The QM cannot be connected to.
Function :	Sender, Server
Severity :	12
Operator Action :	Review System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate sender process for programming error.
System Action :	Fatal error - Communication is terminated.

000052 QUEUE OPEN ERROR

Explanation :	The Server or Sender could not open the associated transmission queue.
Function :	Sender, Server
Severity :	12
Operator Action :	1. 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. 2. Correct problem and restart communication.
System Action :	Fatal error - Communication is terminated.

000053 QUEUE GET ERROR

Explanation :	The Server or Sender could not get a message from the associated transmission queue even if there is (are) message(s) in the transmission queue.
Function :	Sender, Server
Severity :	12
Operator Action :	1. 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. 2. Correct problem and restart communication.
System Action :	Fatal error - Communication is terminated.

000054 QUEUE PUT ERROR

Explanation :	The RECEIVER could not put a message to an application queue.
Function :	Receiver
Severity :	12
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. User action is based upon returned reason code: - Reason code equals MQRC-Q-FULL (2053) or MQRC-Q-SPACE-NOT-AVAILABLE (2056): destination application queue was full and the message was placed on the dead letter queue. Determine if destination queue should be expanded to accommodate more messages or an alternate destination used. All other reason codes: correct problem and restart communication.
System Action :	There are two possible system actions based upon reason code returned from queuer:- 1. Reason code equals MQRC-Q-FULL or MQRC-Q-SPACE-NOT-AVAILABLE: Non-Fatal error - communication will proceed normally after first putting failed put message on dead letter queue. 2. All other reason codes: Fatal error - Communication is terminated.

000055 QUEUE PUT1 ERROR

Explanation :	The RECEIVER could not put a message to the dead letter queue.
Function :	Receiver
Severity :	12
Operator Action :	1. Review the following fields in the error message: QUEUE ID - The 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 problem and restart communication.
System Action :	Fatal error - Communication is terminated.

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. Investigate problem
System Action :	Non-Fatal error - communication will proceed normally. (The unclosed resources, however, will result in a “garbage collection” mechanism be triggered at a proper time to close the unclosed resources).

000057 QUEUE DISC ERROR

Explanation :	An error has occurred to DISCONNECT the connecting Queue Manager.
Function :	Sender, Receiver
Severity :	12
Operator Action :	Review System log or error TD queue for messages prior to this message. Proper running should preclude this occurrence. Investigate sender process for program error.
System Action :	Fatal error - Communication is terminated.

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 terminated because of 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 terminated with error.

005000 CHANNEL CONNECTED

Explanation : Channel connection is successful.

Function : Sender, Receiver

Severity : 0

Operator Action : none

System Action : Platform negotiation will begin.

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 will be opened.

005002 CHANNEL QUEUE OPENED

Explanation : Channel queue has been opened successfully.

Function : Sender, Receiver

Severity : 0

Operator Action : none

System Action : Message transfer will begin.

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 will begin.

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 will be disconnected.

005007 CHANNEL DISCONNECTED

Explanation : Channel has been disconnected from the other platform.

Function : Sender, Receiver

Severity : 0

Operator Action : none

System Action : Channel will be 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)

010000 SYSTEM STARTED W/ ERRORS

Explanation : System being initialized but some queue / channel definition(s) had error(s).

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 definition(s).
3. Shut down and then re-initialize system.

System Action : Erroneous queues / channels are marked as DISABLED.

010001 SYSTEM STARTED W/ FILE ERRORS

Explanation : System being initialized but some queue(s) file(s) had error(s).

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 definition(s).
3. Shut down and then re-initialize system.

System Action : Erroneous queues are marked DISABLED.

010002 SYSTEM STARTED W/ CHANNEL ERRORS

Explanation : System being initialized but some channel definition(s) had error(s).

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 definition(s).
3. Shut down and then re-initialize system.

System Action : Erroneous channel(s) are marked DISABLED.

010003 SYSTEM STARTED BUT SYSTEM CHANGED

Explanation : System being initialized but definitions have been added / deleted while initialization was being performed.

Function : Initialization of system

Severity : 8

Operator Action : Do not perform configuration changes while system is being initialized. Shut down and then re-initialize system.

System Action : If definitions were added then some definition(s) may have been not used.

100000 SYSTEM STOPPED

Explanation : System being stopped while application is running.

Function : System Shutdown

Severity : 0

Operator Action : All applications and channel should be terminated before System is shutdown.

System Action : Terminate request.

100010 SYSTEM ACTIVE

Explanation : System being initialized but System is already active.

Function : Initialization of system

Severity : 0

Operator Action : 1. Shut down System.
2. Re-initialize System.

System Action : System initialization not performed.

100011 SYSTEM STARTED W/ NO QUEUES

Explanation : System being initialized but no queue definitions where found.

Function : Initialization of system

Severity : 4

Operator Action : Add queue definitions and re-initialize 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 re-initialize 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 where found.

Function : Initialization of system

Severity : 12

Operator Action : Delete some channel definitions and re-initialize system.

System Action : System initialized with some Channels.

100090 SYSTEM STARTED W/ NO SYSTEM DEFINITION

Explanation :	System being Initialized but no System Definition was found.
Function :	Initialization of system
Severity :	12
Operator Action :	Define Global System Definition and then initialize the system.
System Action :	System initialization is terminated.

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. Drain this queue either through an application or the queue maintenance facility. 2. Expand the QDEPTH number in the QUEUE definition and refresh this queue's information.
System Action :	PUT request is terminated and the problem queue is marked as “MAX”.

101010 QUEUE CONCURRENT UPDATE HAS OCCURED

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 terminated requests. 2. Re-execute any legitimate requests.
System Action :	The first request is served while the rest of other requests are rejected.

101015 QUEUE NOT FOUND

Explanation :	MQPSSQ, a subroutine to start / stop a queue, reports that the queue to be processed is not defined in the system.
Function :	Start/stop queue
Severity :	8
Operator Action :	Re-execute any terminated requests.
System Action :	The request is terminated unsuccessfully.

101090 QUEUE STOPPED

Explanation : A request has been executed against a STOPPED queue.

Function : Start/stop queue

Severity : 4

Operator Action : START the problem queue

System Action : Terminate the request.

101091 QUEUE DISABLED

Explanation : Queue had errors during initialization.

Function : Initialization of system

Severity : 8

Operator Action : 1. Examine queue definition and file allocation for problem(s).
2. Re-initialize System.

System Action : The problem queue is marked STOPPED.

102090 QUEUE QSN NUMBER LIMIT HAS BEEN REACHED

Explanation : MQPQUE1, a subroutine serving all I/O requests for queues, detects that QSN will exceed the full word limitation of 99,999,999.

Function : General (I/O modules MQPQUE1 and MQPQUE2)

Severity : 8

Operator Action : Perform one of the following :
1. Do file maintenance on this problem queue such as running the batch job MQPREORG.
2. Execute on/line queue maintenance to delete messages via “Delete by Date/time”.

System Action : The PUT request for this queue is rejected.

102091 QUEUE NO SPACE AVAILABLE FOR PUT

Explanation : Queue encounters NOSPACE condition for a PUT request.

Function : General (I/O modules MQPQUE1 and MQPQUE2)

Severity : 8

Operator Action : Perform one of the following :
1. Do file maintenance on this problem queue such as running the batch job MQPREORG.
2. Execute on/line queue Maintenance to delete messages via “Delete by Date/Time”.

System Action : Terminate the request and mark queue “FULL”.

102092 QUEUE NO SPACE AVAILABLE

Explanation : Queue encounters errors for an UPDATE request, NOSPACE condition occurred.

Function :

Severity : 8

Operator Action : Perform one of the following :
1. Do file maintenance on this problem queue such as running the batch job MQPREORG.
2. Execute on/line queue Maintenance to delete messages via “Delete by Date/Time”.

System Action : Terminate the request and mark queue “FULL”.

104021 DUAL QUEUE ERROR

Explanation : Dual destination queue has been STOPPED or was not initialized properly.

Function : General (I/O modules MQPQUE1 and MQPQUE2)

Severity : 8

Operator Action : Perform one of the following :
1. Try to re-START the dual queue.
2. Examine and fix the queue and file definition for this queue. Refresh queue or re-initialize 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 (I/O modules MQPQUE1 and MQPQUE2)

Severity : 8

Operator Action : Perform one of the following :
1. Try to re-START the dual queue.
2. Examine and fix the queue and file definition for this queue. Refresh queue or re-initialize 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 fix the queue and file definition for this queue. Refresh queue or re-initialize system.

System Action : Marked dual queue as “recovery needed”.

105090 QUEUE TRIGGER ERROR

Explanation : MQPSSQ, a subroutine to start / stop a queue, encounters error to start MQ02, a transaction that handles trigger function.

Function : Start/stop queue

Severity : 12

Operator Action : Examine CICS tables to fix the problem.

System Action : The request is terminated unsuccessfully.

105091 QUEUE TRIGGER DATA ERROR

Explanation : MQPAIP2, a program handling trigger function, receives erroneous data and cannot fulfill the request.

Function : Application Interface

Severity : 12

Operator Action : Contact support for MQSeries for VSE.

System Action : The request is terminated unsuccessfully.

109000 ACTION NOT AUTHORIZED

Explanation : NOAUTH condition flagged by CICS when a resource security check has failed.

Function : General (CICS Interface)

Severity : 12

Operator Action : Review security mechanism.

System Action : The request is terminated unsuccessfully.

300000 ACTION NOT SUPPORTED

Explanation : Module has been LINKed to with incorrect function.

Function : General (CICS Interface)

Severity : 12

Operator Action : Review application for call format.

System Action : Terminate 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 : Terminate the request.

300020 PROGRAM HAS REPEATED ERRORS

Explanation : MAPFAIL condition raised in Master Terminal panel(s) (MQMT and its derivatives)

Function : General (CICS Interface)

Severity : 12

Operator Action : Review PPT for MAP modules (MQM????) and fix the problem.

System Action : Terminate the request.

300030 QUEUE LOCK TABLE IS FULL

Explanation : Not enough 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 no application problem is present then increase queue lock count to higher value. Note this value is used to calculate an incore table. So precaution should be used.

System Action : Terminate the request.

301000 EXPECTED RECORD IS MISSING

Explanation : An expected message was found missing. This is normally occurs under a Delete request.

Function : Master Terminal

Severity : 8

Operator Action : Restart the application.

System Action : Terminate the request.

301010 DUPLICATE RECORD HAS OCCURRED

Explanation : An duplicate message was found. This is normally occurs under a PUT condition.

Function : General (MQPQUE1)

Severity : 8

Operator Action : Restart the application.

System Action : Terminate the request.

309010 QUEUE CHECKPOINT RECORD MISSING

Explanation : An checkpoint of a queue was requested and no checkpoint record was found on this queue.

Function : Master Terminal

Severity : 12

Operator Action : Re-initialize system and restart the application.

System Action : Terminate 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 : Terminate the request.

400001 LINK DFHCOMMAREA SIZE INCORRECT

Explanation : Expected DFHCOMMAREA length is incorrect.

Function : General (CICS Interface)

Severity : 12

Operator Action : Examine any prior messages for actual problem.

System Action : Terminate 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 : Terminate 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 : Terminate 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 : Terminate 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 : Terminate the request.

402090 INTERNAL STRUCTURE HAS ERRORS

Explanation : Internal Structure was found corrupted.

Function : General

Severity : 12

Operator Action : Examine any prior messages for actual problem.

System Action : Terminate the request.

501001 CHANNEL FREE ERROR

Explanation : (Reserved)

501002 EIB ERROR

Explanation : RECEIVER encounters an error -
1. upon completion of a GETMAIN command EIBRCODE not equal to low values or
2. upon completion of a RECEIVE command -
RESP not equal to TERMERR and
EIBFREE equal low values and
EIBERR not equal low values.

Function : Receiver

Severity : 8

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 terminated.

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 : 8

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 queuer and corresponding description.
2. Correct problem and restart communication.

System Action : Fatal error - Communication is terminated.

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 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 terminated.

501009 RECEIVER RESPONSES WITH ERROR

Explanation : SENDER receives a rejection from RECEIVER to terminate communication.

Function : Sender

Severity : 8

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 is terminated.

501010 INVALID RESPONSE TYPE

Explanation : Unsupported Message Segment Type received.

Function : Sender

Severity : 8

Operator Action : Review the Segment type and restart communication without the problem type.

System Action : Fatal error - Communication is terminated.

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 counterproposal 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 on initiation.

501020 INVALID FAP LEVEL

Explanation : The protocol in used is not supported.

Function : Sender, Receiver

Severity : 4

Operator Action : none

System Action : Disregard the error and continue on initiation.

501021 MESSAGE SIZE TOO BIG

Explanation :	The message size is too big to be handled.
Function :	Receiver
Severity :	4
Operator Action :	none
System Action :	Suggest a smaller message size and continue on negotiation.

501022 MESSAGE WRAP ERROR

Explanation :	The message sequence number wrap around value cannot be accepted.
Function :	Sender, Receiver
Severity :	4
Operator Action :	none
System Action :	Suggest a smaller value and continue on negotiation.

501023 QUEUE MANGER IS DOWN DURING ACCESSING DLQ

Explanation :	The message is not able to put into the Dead Letter Queue because the System is not up.
Function :	Receiver
Severity :	8
Operator Action :	Initiate System by MQIT or via MQMT.
System Action :	Process is terminated.

501024 QUEUE MANAGER IS DOWN

Explanation :	The communication cannot be established because the Q Manager is down.
Function :	Sender, Receiver
Severity :	8
Operator Action :	Initiate System by MQIT or via MQMT.
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 reports there is an outstanding enqueue on the channel name.

Function : Sender

Severity : 8

Operator Action :
1. Review the following fields in the error message:
CHANNEL ID - channel name that was connected.
2. Determine why second channel was started.
3. Validate channel configuration.

System Action : Fatal error - Communication is terminated.

501028 CHANNEL RE-SYNC ERROR

Explanation : Expected TCF-Confirm-Request flag is 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 encounters -
1. The length of the application portion of the message specified in the header exceeds the maximum length defined for this channel.
2. The length of the application portion of the message received is not equal to the length specified in the header.

Function : Receiver

Severity : 8

Operator Action :
For explanation #1. -
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 insure the maximum message size is set up correctly.
3. Check the configuration of the Sender.
4. Reconfigure if necessary and restart communication.
For explanation #2. -
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 sender/server process for program error.
3. Correct problem and restart communication.

System Action : Fatal error - Communication is terminated.

501031 MESSAGE-PER-BATCH TOO BIG

Explanation :	The maximum number of messages allowed in a batch is too big to be handled.
Function :	Sender, Receiver
Severity :	4
Operator Action :	none
System Action :	Suggest a smaller size and continue on negotiation.

501032 MAX TRANSMISSION SIZE TOO BIG

Explanation :	The maximum transmission size is too big to be handled.
Function :	Sender, Receiver
Severity :	4
Operator Action :	none
System Action :	Suggest a smaller size and continue on negotiation.

501050 RESET MSN

Explanation :	Remote platform MSN (Message Sequence Number) was reset.
Function :	Sender, Receiver
Severity :	4
Operator Action :	none
System Action :	Validate that MSN is within one of this platforms current MSN.

All messages starting with 6000 are severe messages displayed on the CICS terminals from which MQSeries Administrator Dialogs (MQMT) have been started. They 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 correction) the problem persists, please, report this to your IBM support organization.

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 is terminated.

600005 - Prog: xxxxxxxx ABEND Code zzzz Contact Support.

Explanation: The program terminates due to CICS problem and the ABEND code zzzz is returned to an HANDLE ABEND routine.

Severity: 8

Operator Action: Report to IBM

System Action: The dialog is terminated.

600007 - Prog: xxxxxxxx File: yyyyyyy Not Found. Contact Support.

Explanation: A request has been issued against the file yyyyyyyy, but it is 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 is terminated.

600009 - Prog: xxxxxxxx File: yyyyyyy DISABLED. Contact Support.

Explanation: CICS tried to access the file yyyyyyy which was found disabled.

Severity: 8

Operator Action: Use “CEMT S DATA” to set the file ENABLED If the DISABLED status persists, check with the System Administrator. This has nothing to do with MQSeries.

System Action: The dialog is terminated.

600011 - Prog: xxxxxxxx File: yyyyyyy ILLOGIC error. Contact Support.

Explanation: Usually this is related to file I/Os. 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 is terminated.

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 is terminated.

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 is terminated.

600021 - Prog: xxxxxxxx File: yyyyyyy is not open. Contact Support.

Explanation: CICS tried to access a file which was not opened yet, and was unable to open it. This may 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 is terminated.

600023 - Prog: xxxxxxxx INVREQ error Contact Support.

Explanation: An request was received by CICS and may not be processed for various reasons.

Severity: 8

Operator Action: Report to IBM

System Action: The dialog is terminated.

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 is terminated.

600027 - Prog: xxxxxxxx TRANSID error Contact Support.

Explanation: MQseries tried to initiate a transaction, but this transaction was not found in CICS tables.

Severity: 8

Operator Action: This is likely 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 looks good, report the problem to IBM.

System Action: The dialog is terminated.

800000 CICS ERROR CONDITION REACHED

Explanation : ERROR condition of CICS occurred.

Function : General (CICS Interface)

Severity : 12

Operator Action : Investigate the error.

System Action : Terminate 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 : Terminate the request.

800011 ILLOGIC CONDITION

Explanation : ILLOGIC condition of CICS occurred.

Function : General (CICS Interface)

Severity : 12

Operator Action : Investigate the error.

System Action : Terminate 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 : Terminate 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 : Terminate 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 : Terminate 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 : Terminate the request.

802000 NO STORAGE CONDITION

Explanation : A CICS storage is not available.

Function : General (CICS Interface)

Severity : 12

Operator Action : Check that runaway user task has not freeing storage.

System Action : Terminate the request.

803001 LENGTH ERROR CONDITION

Explanation : A record was larger than expected.

Function : General (CICS Interface)

Severity : 12

Operator Action : Check install was done properly.

System Action : Terminate 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 : Terminate 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 : Terminate 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 : Terminate 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 : Terminate the request.

809012 IO ERROR CONDITION

Explanation : An CICS I/O error has occurred.

Function : General (CICS Interface)

Severity : 12

Operator Action : Check CICS log and EIB codes.

System Action : Terminate 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 : Terminate the request.

900000 NO ENVIRONMENT RECORD

Explanation :	Setup of Environment has not been performed.
Function :	Set up system
Severity :	8
Operator Action :	Execute Transaction MQSE to setup Environment.
System Action :	Terminate the request.

Console Messages

Start up messages

The following messages are informational, no operator action required.

MQI0001I -

MQSeries for VSE/ESA starting initialization.

MQI0003I -

MQSeries initialization is complete.

MQI0005I -

FILE : QUEUE : (may have one of the following values:)

not found.

cannot enable.

cannot open.

MQI0011I -

MQSeries for VSE/ESA terminating.

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 B. COBOL programming language examples

This section contains:

- “Language considerations”
- “Calls”
- “Elementary data types”
- “Structure data types”

Language considerations

Copy files

Various COPY files are provided as part of the definition of the message queue interface. These files are used to assist in writing COBOL application programs that use message queuing.

Table 34. Copy files

File name	Contents
CMQGMOV	Get-message options structure
CMQMDV	Message descriptor
CMQODV	Object descriptor
CMQPMOV	Put-message options structure
CMQTMV	Trigger-message structure
CMQV	Named constants

Structures

Each structure declaration begins with a level-10 item. This enables several instances of the structure to be declared, by coding the level-01 declaration and using the **COPY** statement to copy in the remainder of the structure declaration.

The structures should be aligned on 4-byte boundaries. If the **COPY** statement is used to include a structure, following an item which is not the level-01 item, try to ensure that the structure is a multiple of 4-bytes from the start of the level-01 item. Failure to do this may result in a performance degradation.

Notational conventions

The sections that follow show how the:

- Calls should be invoked
- Parameters should be declared
- Various data types should be declared.

In a number of cases, parameters are tables or character strings whose size is not fixed. For these, a lower case “n” is used to represent a numeric constant. When the declaration for that parameter is coded, the “n” must be replaced by the numeric value required.

MQCLOSE

CALL “MQCLOSE” USING HCONN, HOBJ, OPTIONS, COMPCODE, REASON.

Declare the parameters as follows:

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Options that control the action of MQCLOSE
01 OPTIONS    PIC S9(9) BINARY.
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

MQCONN

CALL “MQCONN” USING NAME, HCONN, COMPCODE, REASON.

Declare the parameters as follows:

```
** Name of queue manager
01 NAME    PIC X(48).
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

MQDISC

CALL “MQDISC” USING HCONN, COMPCODE, REASON.

Declare the parameters as follows:

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON      PIC S9(9) BINARY.
```

MQGET

CALL “MQGET” USING HCONN, HOBJ, MSGDESC, GETMSGOPTS, BUFFERLENGTH, BUFFER, DATALENGTH, COMPCODE, REASON.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Message descriptor
01 MSGDESC.
 COPY MQIMQMD.
** Options that control the action of MQGET
01 GETMSGOPTS.
 COPY MQIMQGM.
** Length in bytes of the Buffer area
01 BUFFERLENGTH    PIC S9(9) BINARY.
** Area to contain the message data
01 BUFFER    PIC X(n).
** Length of the message
```

```
01 DATALENGTH    PIC S9(9) BINARY.
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

MQINQ

CALL “MQINQ” USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE, INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH, CHARATTRS, COMPCODE, REASON.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Count of selectors
01 SELECTORCOUNT     PIC S9(9) BINARY.
** Array of attribute selectors
01 SELECTORS-TABLE
02 SELECTORS     PIC S9(9) BINARY OCCURS n TIMES.
** Count of integer attributes
01 INTATTRCOUNT     PIC S9(9) BINARY.
** Array of integer attributes
01 INTATTRS-TABLE
02 INTATTRS     PIC S9(9) BINARY OCCURS n TIMES.
** Length of character attributes buffer
01 CHARATTRLENGTH     PIC S9(9) BINARY.
** Character attributes
01 CHARATTRS    PIC X(n).
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

MQOPEN

CALL “MQOPEN” USING HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object descriptor
01 OBJDESC.
 COPY MQIMQOD.
** Options that control the action of MQOPEN
01 OPTIONS            PIC S9(9) BINARY.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

MQPUT

CALL “MQPUT” USING HCONN, HOBJ, OBJDESC, MSGDESC, PUTMSGOPTS, BUFFERLENGTH, BUFFER, COMPCODE, REASON.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Message descriptor
01 MSGDESC.
```

```
 COPY MQIMQMD.
** Options that control the action of MQPUT
01 PUTMSGOPTS.
 COPY MQIMQPM.
** Length of the message in Buffer
01 BUFFERLENGTH    PIC S9(9) BINARY.
** Message data
01 BUFFER    PIC X(n).
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

MQPUT1

CALL “MQPUT1” USING HCONN, OBJDESC, MSGDESC, PUTMSGOPTS, BUFFERLENGTH, BUFFER, COMPCODE, REASON.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object descriptor
01 OBJDESC.
** Message descriptor
01 MSGDESC.
 COPY MQIMQMD.
** Options that control the action of MQPUT
01 PUTMSGOPTS.
 COPY MQIMQPM.
** Length of the message in Buffer
01 BUFFERLENGTH    PIC S9(9) BINARY.
** Message data
01 BUFFER    PIC X(n).
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying CompCode
01 REASON     PIC S9(9) BINARY.
```

Elementary data types

Table 35. Elementary data types

Data type	Representation
MQBYTE	PIC X
MQBYTE24	PIC X(24)
MQBYTE32	PIC X(34)
MQCHAR	PIC X
MQCHAR4	PIC X(4)
MQCHAR8	PIC X(8)
MQCHAR12	PIC X(12)
MQCHAR28	PIC X(28)
MQCHAR32	PIC X(32)
MQCHAR48	PIC X(48)
MQCHAR64	PIC X(64)
MQCHAR128	PIC X(128)
MQCHAR256	PIC X(256)
MQHCONN	PIC S(9) BINARY
MQHOBJ	PIC S(9) BINARY
MQLONG	PIC S(9) BINARY

Structure data types

MQGMO in Copybook CMQGMOV

```
** MQGMO structure
10 MQGMO.
** Structure identifier
15 MQGMO-STRUCID    PIC X(4).
** Structure version number
15 MQGMO-VERSION    PIC S9(9) BINARY.
** Options
15 MQGMO-OPTIONS    PIC S9(9) BINARY.
** Wait interval
15 MQGMO-WAIT-INTERVAL      PIC S9(9) BINARY.
** Signal
15 MQGMO-SIGNAL1    PIC S9(9) BINARY.
** Reserved
15 MQGMO-SIGNAL2    PIC S9(9) BINARY.
** Resolved name of destination queue
15 MQGMO-RESOLVED-QNAME        PIC S9(9) BINARY.
```

MQMD in Copybook CMQMDV

```
** MQMD structure
10 MQMD.
** Structure identifier
15 MQMD-STRUCID    PIC X(4).
** Structure version number
15 MQMD-VERSION    PIC S9(9) BINARY.
** Reports
15 MQMD-REPORT    PIC S9(9) BINARY.
** Message type
15 MQMD-MSGTYPE    PIC S9(9) BINARY.
** Reserved
15 MQMD-EXPIRY    PIC S9(9) BINARY.
** Feedback code
15 MQMD-FEEDBACK    PIC S9(9) BINARY.
** Data encoding
15 MQMD-ENCODING     PIC S9(9) BINARY.
** Coded character set identifier
15 MQMD-CODED-CHAR-SET-ID    PIC S9(9) BINARY.
** Format name
15 MQMD-FORMAT    PIC X(8).
** Message priority
15 MQMD-PRIORITY    PIC S9(9) BINARY.
** Message persistence
15 MQMD-PERSISTENCE    PIC S9(9) BINARY.
** Message identifier
15 MQMD-MSGID    PIC X(24).
** Correlation identifier
15 MQMD-CORRELID    PIC X(24).
** Backout counter
15 MQMD-BACKOUT-COUNT    PIC S9(9) BINARY.
** Name of reply-to queue
15 MQMD-REPLYTOQ    PIC X(48).
** Name of reply queue manager
15 MQMD-REPLY-TO-QMGR    PIC X(48).
** User identifier
15 MQMD-USERIDENTIFIER    PIC X(12).
** Accounting token
15 MQMD-ACCOUNTING-TOKEN    PIC X(32).
** Application data relating to identity
15 MQMD-APPL-IDENTITY-DATA    PIC X(32).
** Type of application that put the message
15 MQMD-PUT-APPL-TYPE    PIC S9(9) BINARY.
** Name of application that put the message
15 MQMD-PUT-APPL-NAME    PIC X(28).
** Date when message was put
15 MQMD-PUTDATE     PIC X(8).
** Time (GMT) when message was put
15 MQMD-PUTTIME     PIC X(8).
** Application data relating to origin
15 MQMD-APPL-ORIGIN-DATA    PIC X(4).
```

MQOD in Copybook CMQODV

```
** MQOD structure
10 MQOD.
** Structure identifier
15 MQOD-STRUCID    PIC X(4).
** Structure version number
15 MQOD-VERSION    PIC S9(9) BINARY.
** Object type
15 MQOD-OBJECTTYPE    PIC S9(9) BINARY.
** Object name
15 MQOD-OBJECTNAME    PIC X(48).
** Object queue manager name
15 MQOD-OBJECTQMGRNAME    PIC X(48).
** Dynamic queue name
15 MQOD-DYNAMICQNAME    PIC X(48).
** Alternate user identifier
15 MQOD-ALTERNATEUSERID    PIC X(12).
```

MQPMO in Copybook CMQPMOV

```
** MQPMO structure
10 MQPMO.
** Structure identifier
15 MQPMO-STRUCID    PIC X(4).
** Structure version number
15 MQPMO-VERSION    PIC S9(9) BINARY.
** Options
15 MQPMO-OPTIONS    PIC S9(9) BINARY.
** Reserved
15 MQPMO-TIMEOUT    PIC S9(9) BINARY.
** Object handle of input queue
15 MQPMO-CONTEXT    PIC S9(9) BINARY.
** Reserved
15 MQPMO-KNOWNDESTCOUNT    PIC S9(9) BINARY.
** Reserved
15 MQPMO-UNKNOWNDESTCOUNT    PIC S9(9) BINARY.
** Reserved
15 MQPMO-INVALIDDESTCOUNT    PIC S9(9) BINARY.
** Resolved name of destination queue
15 MQPMO-RESOLVEDQNAME    PIC X(48).
** Resolved name of destination queue manager
15 MQPMO-RESOLVEDQMGRNAME    PIC X(48).
```

Appendix C. CICS control table definitions

The following sub-appendices contain sample entries for the CICS control tables, including:

- File Control Table (FCT)
- Destination Control Table (DCT)
- Programs and Transactions

Sample FCT entries

Note: Entry named MQFCNFG is required.

```
*-------------------------------------------------------------------*
* Licensed Materials - Property of IBM                              *
*                                                                   *
* 5787-ECX                                                          *
* (C) Copyright IBM Corp. 1993, 1996                                *
*                                                                   *
* US Government Users Restricted Rights - Use, duplication or       *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
*-------------------------------------------------------------------*
*
*-------------------------------------------------------------------*
*         Start of  MQ/Series VSAM cluster definitions              *
*                                                                   *
*  For performance reasons entries may be modified to add LSRPOOL   *
*  explicit specifications.                                         *
*-------------------------------------------------------------------*
*
* system ssetup file
MQFSSET  DFHFCT TYPE=DATASET,DATASET=MQFSSET,                        *
               ACCMETH=VSAM,                                        *
               SERVREQ=(READ,BROWSE),                               *
               LOG=NO,                                              *
               RSL=PUBLIC,                                          *
               BUFND=5,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,                                          *
               BUFND=5,BUFNI=10,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,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                          *
               RECFORM=(VARIABLE,BLOCKED)
MQFO001  DFHFCT TYPE=DATASET,DATASET=MQFO001,                        *
               ACCMETH=VSAM,                                        *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),             *
               LOG=YES,                                             *
               RSL=PUBLIC,                                          *
               BUFND=16,BUFNI=16,STRNO=16,                          *
```

```
               RECFORM=(VARIABLE,BLOCKED)
MQFI002  DFHFCT TYPE=DATASET,DATASET=MQFI002,                          *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               RSL=PUBLIC,                                             *
               LOG=YES,                                                *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
MQFO002  DFHFCT TYPE=DATASET,DATASET=MQFO002,                          *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               LOG=YES,                                                *
               RSL=PUBLIC,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
MQFI003  DFHFCT TYPE=DATASET,DATASET=MQFI003,                          *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               LOG=YES,                                                *
               RSL=PUBLIC,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
MQFO003  DFHFCT TYPE=DATASET,DATASET=MQFO003,                          *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               LOG=YES,                                                *
               RSL=PUBLIC,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
*--SYSTEM DEFINITIONS
MQFLOG   DFHFCT TYPE=DATASET,DATASET=MQFLOG,                           *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               RSL=PUBLIC,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
MQFERR   DFHFCT TYPE=DATASET,DATASET=MQFERR,                           *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               RSL=PUBLIC,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
MQFMON   DFHFCT TYPE=DATASET,DATASET=MQFMON,                           *
               ACCMETH=VSAM,                                           *
               SERVREQ=(READ,UPDATE,ADD,BROWSE,DELETE),                *
               RSL=PUBLIC,                                             *
               BUFND=16,BUFNI=16,STRNO=16,                             *
               RECFORM=(VARIABLE,BLOCKED)
*-----------------------------------------------------------------------*
*            End  of MQ/Series VSAM cluster definitions                 *
*-----------------------------------------------------------------------*
```

Sample DCT entry

Note: Entry named MQER is required in order for MQSeries System error messages to be logged to the SYSTEM.LOG queue.

```
*--------------------------------------------------------------------*
* Licensed Materials - Property of IBM                               *
*                                                                    *
* 5787-ECX                                                           *
* (C) Copyright IBM Corp. 1993, 1996                                 *
*                                                                    *
* 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 execute 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                               *
*                                                                    *
* 5787-ECX                                                           *
* (C) Copyright IBM Corp. 1993, 1996                                 *
*                                                                    *
* 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                              *
*                                                                   *
* 5787-ECX                                                          *
* (C) Copyright IBM Corp. 1993, 1996                                *
*                                                                   *
* 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 CICSFCT*
*   therefore if there are any new file ids to be added in here,    *
*   it must also be added into the corresponding JCL                *
*                                                                   *
*   Fields filed with ?volid? have to be modified to suit the user  *
*   specifications.                                                 *
*-------------------------------------------------------------------*
// DLBL MQFSSET,'MQSERIES.MQFSSET',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFCNFG,'MQSERIES.MQFCNFG',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFIO01,'MQSERIES.MQFIO01',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFIO02,'MQSERIES.MQFIO02',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFIO03,'MQSERIES.MQFIO03',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFOO01,'MQSERIES.MQFOO01',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFOO02,'MQSERIES.MQFOO02',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFOO03,'MQSERIES.MQFOO03',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFERR,'MQSERIES.MQFERR',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFLOG,'MQSERIES.MQFLOG',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
// DLBL MQFMON,'MQSERIES.MQFMON',0,VSAM,CAT=MQMCAT
// EXTENT ,?volid?
*-------------------------------------------------------------------*
*   End of sample jcl file definition for cics deck                 *
*-------------------------------------------------------------------*
```

Sample JCL to create CICS CSD group

```
* ** JOB JNM=MQJCSD,CLASS=O,DISP=D
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJCSD Define resources for MQ/Series for VSE/ESA to CICS CSD.
* ------------------------------------------------------------------*   00001300
*   Please change :                                                 *
*                 "*  ** JOB" to  "* $$ JOB"                        *
*                 "*  ** LST" to  "* $$ LST"                        *
*                 "*  ** EOJ" to  "* $$ EOJ"                        *
* ------------------------------------------------------------------*   00001300
*       Create CICS CSD group for MQ/Series VSE/ESA                 *
*                                                                   *
*   This file is a sample and may need modifications to suit the    *
*   users environment (eg. Group name, or list name).               *
* ------------------------------------------------------------------*   00001300
* Licensed Materials - Property of IBM                              *
*                                                                   *
* 5787-ECX                                                          *
* (C) Copyright IBM Corp. 1993, 1996                                *
*                                                                   *
* US Government Users Restricted Rights - Use, duplication or       *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
* ------------------------------------------------------------------*   00001300
// EXEC DFHCSDUP                                                        00002110
*-------------------------------------------------------------------*   00002100
*            Definitions for MQ/Series VSE/ESA                      *
*-------------------------------------------------------------------*   00002100
*                                                                       00002200
DELETE GROUP(MQM)                                                       00056000
*
*--                     Definitions of MQ/Series Programs
*
 DEFINE  PROGRAM(MQPMTP  ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPMCFG ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPMMON ) 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(MQPMSI  ) 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(MQPMMOC ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
*-- NON-ADMINISTRATOR
 DEFINE  PROGRAM(MQPAIPO ) 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(MQPCCKPT) 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(MQPSCHK ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPERR  ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPFINDQ) 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(MQPSMAP ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPSSET ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPSENV ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
 DEFINE  PROGRAM(MQPCMD  ) GROUP(MQM) LANGUAGE(COBOL)     RSL(PUBLIC)
*-- MAPS
 DEFINE  PROGRAM(MQMMTP  ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMCFG ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMMON ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMOPR ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMDISP ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMSYS ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMQUE ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMCHN ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMSS  ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMSC  ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMMSN ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMSI  ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMDEL ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMMOQ ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
 DEFINE  PROGRAM(MQMMMOC ) GROUP(MQM) LANGUAGE(ASSEMBLER) 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  PROGRAM(TTMTST3 ) GROUP(MQM) LANGUAGE(ASSEMBLER) RSL(PUBLIC)
*
*--                     Definitions of MQ/Series Transactions
*
 DEFINE TRANSACTION(MQMT)  GROUP(MQM) PROGRAM(MQPMTP)
 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(MQPMMOC)
 DEFINE TRANSACTION(MQIT)  GROUP(MQM) PROGRAM(MQPINIT1)
 DEFINE TRANSACTION(MQO2)  GROUP(MQM) PROGRAM(MQPAIP2)
 DEFINE TRANSACTION(MQO1)  GROUP(MQM) PROGRAM(MQPRECV)
 DEFINE TRANSACTION(MQO3)  GROUP(MQM) PROGRAM(MQPSEND)
 DEFINE TRANSACTION(MQSS)  GROUP(MQM) PROGRAM(MQPSSQ)
```

```
 DEFINE TRANSACTION(MQSM)  GROUP(MQM) PROGRAM(MQPSCHK)
 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(MQPSSET)
 DEFINE TRANSACTION(MQSE)  GROUP(MQM) PROGRAM(MQPSENV)
 DEFINE TRANSACTION(MQSR)  GROUP(MQM) PROGRAM(MQPSREC)
 DEFINE TRANSACTION(MQSQ)  GROUP(MQM) PROGRAM(MQPQREC)
*-- Test Transactions
 DEFINE TRANSACTION(TST1)  GROUP(MQM) PROGRAM(TTPTST1)
 DEFINE TRANSACTION(TST2)  GROUP(MQM) PROGRAM(TTPTST2)
 DEFINE TRANSACTION(TST3)  GROUP(MQM) PROGRAM(TTPTST3)
*-- Add MQ/Series group to the standard VSE/ESA list.
 ADD GROUP(MQM) LIST(VSELIST)
/*
/&
* ** EOJ
```

Programs and transactions

The following programs and transactions must be defined for CICS using PCT and PPT entries or via RDO.

BMS maps

```
MQMDISP, MQMMDEL, MQMMCFG, MQMMCHN, MQMMMOC, MQMMMON,
MQMMMOQ, MQMMMSN, MQMMOPR, MQMMQUE, MQMMSC, MQMMSI,
MQMMSS, MQMMSYS, MQMMTP, TTMTST3.
```

COBOL for VSE programs and transactions

TRANIDs	Program	Description
MQMT	MQPMTP	Master Terminal Main Program
MQMC	MQPMCFG	" " Configuration Main Program
MQMQ/MQDQ	MQPMQUE	" " Queue definition program.
MQMS/MQDS	MQPMSYS	" " Global System definition.
MQMH/MQDH	MQPMCHN	" " Channel definition.
MQMO	MQPMOPR	" " Operations Main program.
MQMA	MQPMSS	" " Start/Stop Queue
MQMB	MQPMSC	" " Open/Close Channel
MQMR	MQPMMSN	" " Message Reset
MQMI	MQPMSI	" " Initialize System
MQMM	MQPMMON	" " Monitor Main program
MQQM	MQPMMOQ	" " Monitor Queue
MQCM	MQPMMOC	" " Monitor Channel
MQMD	MQPMDEL	" " Queue Records Maintenance
	MQPSTART	PLT post initialization transaction to invoke transaction MQIT
MQIT	MQPINIT1	System Initialization
	MQPINIT2	Channel Initialization
MQSS	MQPSSQ	Start/Stop Queue
	MQPAIPO	Application Interface Program (AIP) for MQM Stub's and System AIP.
	MQPAIP1	System AIP.
	MQPQUE1	Queue Manager
	MQPQUE2	Internal Queue Manager
MQQD/MQQA	MQPQDEL	Update of queue records
MQER	MQPERR	System error program
	MQPECHO	Test ECHO program (see page 255)
MQSE	MQPSENV	Setup environment
MQSM	MQPSCHK	System Monitor
	MQPFINDQ	Performs Queue/Queue Manager lookup
MQST	MQPSTOP	System Shutdown Task
MQSU	MQPSSET	Setup System configuration file
MQO1	MQPRECV	Channel Receiver
MQO2	MQPAIP2	Trigger AIP Handler
MQO3	MQPSEND	Channel Sender
	MQPCCKPT	Channel Checkpoint
TST1	TTPTST1	Test 1 program (see page 207)
TST2	TTPTST2	Test 2 program (see page 221)
TST3	TTPTST3	Test 3 program (see page 239)

Appendix D. Sample programs

Sample program TTPTST1.Z

This program is a test facility for sending/receiving messages. It must be invoked by terminal input format as:

```
TST1 func nn queue-name
```

Where:

TST1 is the transaction id
func is any of the following functions:

BOTH	put and get message(s)
GET	get message(s)
GETD	get and delete message(s)
INQ	invoke MQINQ about queue's attributes
PUT	put message(s)
PUTR	put message(s) and send reply
PUT1	put and delete messages

nn is the number of messages to be processed (01 through 99)
queue-name is the name of the local or transmission queue to be processed.

For example, "TST1 PUT 99 QUE1" will put 99 messages into a transmission queue named QUE1 (the messages will read "THIS IS A MESSAGE TEXT"). Typing TST1 alone will display help instructions.

For each function, there is a corresponding set of `MQCONN, MQOPEN, MQCLOSE` and `MQDISC`. In the above example, there would be 99 connections and disconnections to the Queue Manager and 99 opens and closes to QUE1.

```
*/INCLUDE COPYRSAP
*****************************************************************
* Licensed Materials - Property of IBM                          *
*                                                               *
* 5787-ECX                                                      *
* (C) Copyright IBM Corp. 1993, 1996                            *
*                                                               *
* US Government Users Restricted Rights - Use, duplication or   *
* disclosure restricted by GSA ADP Schedule Contract with IBM   *
* Corp.                                                         *
*****************************************************************
       IDENTIFICATION DIVISION.
       PROGRAM-ID.    TTPTST1.
       AUTHOR.        IBM.

       DATE-WRITTEN.  12/15/92.
       DATE-COMPILED.
      *LAST-MODIFIED.  3/21/96.

*****************************************************************
*---------------------------------------------------------------*
*---------------------------------------------------------------*
*               T E S T                                         *
*                                                               *
*          A P P L I C A T I O N   I N T E R F A C E            *
*                                                               *
*                 IBM MQI                                       *
*                                                               *
*---------------------------------------------------------------*
* TTPTST1 - MQI APPLICATION TEST PROGRAM                        *
*                                                               *
* FUNCTIONS:  1. PERFORM NORMAL QUEUE PUT                       *
*             2. TRY TO GET QUEUE INFO BACK                     *
*                                                               *
* COPYBOOKS:  MQIVALUE - MQI RETURN CODES.                      *
*                                                               *
* 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(80) VALUE
            'Licensed Materials - Property of IBM'.
            05 FILLER                 PIC X(80) VALUE SPACES.
            05 FILLER                 PIC X(80) VALUE
            '5787-ECX  '.
            05 FILLER                 PIC X(80) VALUE SPACES.
            05 FILLER                 PIC X(80) VALUE
            '(C) Copyright IBM Corp. 1993, 1996   All Rights
                 Reserved'.
```

```
           05 FILLER                  PIC X(80) VALUE SPACES.
           05 FILLER                  PIC X(80) VALUE
           'US Government Users Restricted Rights - Use,
                duplication '.
           05 FILLER                  PIC X(80) VALUE
          'or disclosure restricted by GSA ADP Schedule Contract
                '.
           05 FILLER                  PIC X(80) VALUE
           'with IBM Corp.'.

*-------------------------------------------------------------*

       01  FILLER                        PIC X(40) VALUE
                     'TTPTST1 WORKING STORAGE STARTS HERE ==>'.

       01  WS-VERSION.
           05  FILLER                  PIC X(30)      VALUE
           'TTPTST1 VERSION 1.4'.

       01  WS-WORK-FIELDS.
          05  WS-IDX                  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-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-YY              PIC  99  VALUE ZERO.
              10  WS-DATE-MM              PIC  99  VALUE ZERO.
              10  WS-DATE-DD              PIC  99  VALUE ZERO.
           05  WS-TIME-9                  PIC  9(7)  VALUE ZERO.
           05  WS-TIME.
              10  FILLER                  PIC  9   VALUE ZERO.
              10  WS-TIME-HH              PIC  99  VALUE ZERO.
              10  WS-TIME-MM              PIC  99  VALUE ZERO.
              10  WS-TIME-SS              PIC  99  VALUE ZERO.

           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-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
*-------------------------------------------------------------*
*-------------------------------------------------------------*
       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  FILLER             PIC  X(5)  VALUE 'TST1 '.
                  15  WS-DATA-FUNCTION    PIC  XXXX  VALUE 'PUT'.
                   88 WS-PUT                        VALUE 'PUT'.
                   88 WS-INQ                        VALUE 'INQ'.
                   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'.

                12  FILLER               PIC  X     VALUE ' '.
                   12  WS-DATA-TIMES         PIC  99    VALUE zeros.
                10  FILLER                  PIC  X     VALUE ' '.
                10  WS-DATA-QUEUE           PIC  X(48) VALUE SPACES.

        EJECT
*-------------------------------------------------------------*
       01  WS-NEED-REPLY.
           05  FILLER                  PIC  X(80) VALUE
           'Please enter REPLY QUEUE name with trailing blanks or
                ErsEOF
      -    ' (e.g. Ctrl - Del)'.
        EJECT
*-------------------------------------------------------------*
       01  WS-HELP.
           05  FILLER                  PIC  X(80) VALUE
           ' TST1 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
           ' TST1 XXXX NN
                QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ
      -    '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
           '                      PUT  - MQPUT  MESSAGES'.
           05  FILLER                  PIC  X(80) VALUE
           '                      PUT1 - MQPUT1 MESSAGES'.
           05  FILLER                  PIC  X(80) VALUE
           '                      INQ  - MQINQ ALL INFO.'.
           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
           '  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 16 TIMES
                                       PIC  X(80).

*-------------------------------------------------------------*
        EJECT
*-------------------------------------------------------------*
       01  WS-OK-MSG.
           05 WS-OK-MSG-0               PIC  X(80) VALUE
            ' FULL CYCLE HAS BEEN PERFORMED SUCCESSFULLY'.
           05 WS-OK-MSG-1               PIC  X(80) VALUE SPACES.
           05 WS-OK-MSG-2               PIC  X(80) VALUE SPACES.
           05 WS-OK-MSG-3               PIC  X(80) VALUE SPACES.
           05 WS-OK-MSG-4               PIC  X(80) VALUE SPACES.
           05 WS-OK-MSG-5               PIC  X(80) VALUE SPACES.
           05 WS-OK-MSG-6               PIC  X(80) VALUE SPACES.

       01  WS-INVALID-MSG.
           05  FILLER                  PIC  X(40) VALUE
               'QUEUE NAME MISSING, PROCESS TERMINATED'.

       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).

*-----------------------------------------------------------------*
        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.
      * COPY CMQV.
      */INCLUDE CMQV
      */INCLUDE  COPYR
****************************************************************
**                                                            **
**  FILE NAME:        CMQV                                    **
**                                                            **
**  DESCRIPTIVE NAME: COBOL copy file for MQI constants       **
**                                                            **
**  VERSION 1.4.0                                           **
**                                                            **
**  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-SYNCPOINT               PIC S9(9) BINARY  VALUE 2.
          10 MQGMO-NO-SYNCPOINT            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-INTEGER-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-INTEGER-UNDEFINED PIC S9(9) BINARY  VALUE 0.
        10 MQENC-INTEGER-NORMAL    PIC S9(9) BINARY  VALUE 1.
        10 MQENC-INTEGER-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.

**   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-SYNCPOINT            PIC S9(9) BINARY  VALUE 2.
        10 MQPMO-NO-SYNCPOINT         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-SYNCPOINT-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
*--------------------------------------------------------------*
* 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-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.
*    COPY  CMQODV.
*/INCLUDE CMQODV
*/INCLUDE  COPYR
****************************************************************
**                                                            **
**  FILE NAME:        CMQODV                                  **
**                                                            **
**  DESCRIPTIVE NAME: COBOL copy file for MQOD structure      **
**                                                            **
**  VERSION 1.4.0                                           **
**                                                            **
**  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


*--PUT/GET PARM
       01  WS-MSG-DESCRIPTOR.
*       COPY CMQMDV.
*/INCLUDE CMQMDV
*/INCLUDE  COPYR
****************************************************************
```

```
**                                                               **
**  FILE NAME:        CMQMDV                                     **
**                                                               **
**  DESCRIPTIVE NAME: COBOL copy file for MQMD structure         **
**                                                               **
**  VERSION 1.4.0                                              **
**                                                               **
**  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.
*/INCLUDE CMQPMOV
*/INCLUDE  COPYR
*****************************************************************
**                                                               **
**  FILE NAME:        CMQPMOV                                    **
**                                                               **
**  DESCRIPTIVE NAME: COBOL copy file for MQPMO structure        **
**                                                               **
**  VERSION 1.4.0                                              **
**                                                               **
**  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-KNOWNDESTCOUNT   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.
*    COPY   CMQGMOV.
*/INCLUDE CMQGMOV
*/INCLUDE  COPYR
*****************************************************************
**                                                               **
**  FILE NAME:        CMQGMOV                                    **
**                                                               **
**  DESCRIPTIVE NAME: COBOL copy file for MQGMO structure        **
**                                                               **
**  VERSION 1.4.0                                              **
**                                                               **
**  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.

       01  WS-BUFFER-AREA.
           05  FILLER                     PIC  X(500) VALUE
           'THIS IS A MESSAGE TEXT'.

       01  WS-ORIGINAL-BUFFER-AREA.
           05  FILLER                     PIC  X(200) VALUE
           'THIS IS A MESSAGE TEXT'.

*--INQ FIELDS
       01  MQI-SECTOR-COUNT            PIC S9(8) COMP VALUE ZERO.

       01  MQI-SECTOR.
           05  MQI-SECTOR-ENTRY   OCCURS 40 TIMES
                                       PIC S9(8) COMP.

       01  MQI-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.

       01  MQI-CHAR-ATTR-LENGTH        PIC S9(8) COMP VALUE +500.

       01  MQI-CHAR-ATTR.
           05  FILLER                  PIC X(500) VALUE SPACES.

*-------------------------------------------------------------*
        EJECT
*-------------------------------------------------------------*
       LINKAGE SECTION.
*-------------------------------------------------------------*
       01  DFHCOMMAREA.
           05  FILLER                    PIC X.

       01  LK-DATA.
           05  FILLER                    PIC X(1000).
        EJECT
*-------------------------------------------------------------*
       PROCEDURE DIVISION.
*-------------------------------------------------------------*

       0000-MAIN-LINE.

*--INITIALIZE
           MOVE  'INIT   ' TO   WS-LEVEL.
           PERFORM 1000-INITIALIZE
              THRU 1000-EXIT.

*-------------------------------------------------------------*
           PERFORM   WS-PROCESS-TIMES   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 7000-INQ-MESSAGES
                        THRU 7000-EXIT
                END-IF

*-- --IF NO MORE MESSAGES ...GET OUT
                IF WS-END-OF-MESSAGES
                 THEN
                       GO TO  0000-ENDIT
                END-IF

           END-PERFORM.
*-------------------------------------------------------------*
*--GET DURACTION TIME
       0000-ENDIT.
           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-PROCESS-TIMES  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
               MOVE  WS-REPLY-Q        TO  WS-OK-QUEUE-REPLY
               MOVE WS-OK-STATS-LINE-2 TO  WS-OK-MSG-2
           END-IF.
           IF  WS-OK-QUEUE EQUAL SPACES   OR
               (WS-PUT-WITH-REPLY         AND
                WS-OK-QUEUE-REPLY EQUAL SPACES)
               MOVE  ZEROS  TO   WS-OK-MESSAGES
               MOVE  WS-INVALID-MSG TO WS-OK-MSG-0
           END-IF.

*-- --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.
*

           EXEC CICS  SEND
                      FROM   (WS-OK-MSG)
                      LENGTH (LENGTH OF WS-OK-MSG)
```

```
                    ERASE
           END-EXEC.

*-------------------------------------------------------------*
       0000-RETURN.
           EXEC CICS RETURN
                END-EXEC.

           GOBACK.
        EJECT
*-------------------------------------------------------------*
       1000-INITIALIZE.
*-------------------------------------------------------------*
* PURPOSE: SETUP DATA AREAS
*-------------------------------------------------------------*
           EXEC  CICS ASKTIME
                      ABSTIME(WS-ABSTIME)
                 END-EXEC.
*
           EXEC  CICS FORMATTIME
                      ABSTIME(WS-ABSTIME)
                      YYMMDD (WS-DATE)
                 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.
*
*--GET INPUT INFO...
           EXEC  CICS RECEIVE
                            SET(ADDRESS OF LK-DATA)
                           LENGTH(WS-DATA-LENGTH)
                END-EXEC.

*--CHECK WHAT WE'RE DOING
*-- --COMMAND IS  "TST1 GET 01 QUEUENAME"
           IF (WS-DATA-LENGTH < LENGTH OF WS-DATA-WITH-FUNCTION)
           OR (WS-DATA-LENGTH > LENGTH OF WS-DATA-ALL)
            THEN
               PERFORM  1100-SEND-HELP
               GO TO   0000-RETURN.

*--DO VARIABLE MOVE
           CALL 'MQPMOVE'  USING  WS-DATA-WITH-QUEUE
                                  LK-DATA
                                  WS-DATA-LENGTH.

*
           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.

*--DO VARIABLE MOVE
           CALL 'MQPMOVE'  USING  WS-REPLY-Q
                                  LK-DATA
                                  WS-DATA-LENGTH.

*

*-------------------------------------------------------------*
       1000-EXIT.
           EXIT.
         EJECT
*-------------------------------------------------------------*
       1100-SEND-HELP.
*-------------------------------------------------------------*
*--SEND HELPLIST
           EXEC CICS  SEND
                      FROM   (WS-HELP)
                      LENGTH (LENGTH OF WS-HELP)
                      ERASE
           END-EXEC.

*-------------------------------------------------------------*
        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.

*
*--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.
*WKH
          EXEC  CICS SYNCPOINT
             END-EXEC.
*
*--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.

*
*--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  SPACES    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
                   ELSE
                       GO TO  9900-ERR-DISPLAY.
*

*--ADDED 4/ 5/93
          EXEC  CICS SYNCPOINT
             END-EXEC.
*
*--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.
*
*--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.
*

*--ADDED 4/ 5/93
           EXEC  CICS SYNCPOINT
              END-EXEC.

*--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 MQOO-BROWSE       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.

*
*--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  SPACES    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
                  ELSE
                      GO TO  9900-ERR-DISPLAY.
*

*--MQGET TO QUEUE TO QM W/ DELETE UNDER CURSOR
           IF WS-CCODE-VALUE    EQUAL  ZERO
            THEN
                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  SPACES    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
                             NEXT SENTENCE
                         ELSE
                             GO TO  9900-ERR-DISPLAY.
*

*--ADDED 4/ 5/93
           EXEC  CICS SYNCPOINT
              END-EXEC.
*
*--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 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 REPLY QUEUE
           MOVE  MQMT-REPLY   TO  MQMD-MSGTYPE.
           MOVE  SPACES       TO  MQMD-REPLYTOQMGR.
           MOVE  WS-REPLY-Q   TO  MQMD-REPLYTOQ.

           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.

*
*--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.

*--ADDED 4/ 5/93
```

```
          EXEC  CICS SYNCPOINT
             END-EXEC.
*
*--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.

*-------------------------------------------------------------*
      6000-EXIT.
          EXIT.
       EJECT
*-------------------------------------------------------------*
      7000-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 MQOO-INQUIRE   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.

*--SETUP INQ PARMS
          MOVE MQCA-Q-DESC   TO MQI-SECTOR-ENTRY (1).
          MOVE MQCA-Q-NAME   TO MQI-SECTOR-ENTRY (2).
          MOVE MQIA-INHIBIT-PUT   TO MQI-SECTOR-ENTRY (3).
          MOVE MQIA-Q-TYPE   TO MQI-SECTOR-ENTRY (4).
          MOVE MQIA-MAX-MSG-LENGTH   TO MQI-SECTOR-ENTRY (5).
          MOVE +5      TO MQI-SECTOR-COUNT.
*
*--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
                                  MQI-SECTOR-COUNT
                                  MQI-SECTOR
                                  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.
*
*--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.

*-------------------------------------------------------------*
      7000-EXIT.
          EXIT.
        EJECT
*-------------------------------------------------------------*
      9900-ERR-DATA.
*-------------------------------------------------------------*
*--ERROR IN "GET" DATA
          MOVE  WS-DATA-LENGTH-USER  TO  WS-ERR-DATA-LENGTH.
          MOVE  WS-BUFFER-AREA       TO  WS-ERR-DATA-AREA.
          EXEC CICS  SEND
                    FROM   (WS-ERR-DATA)
                    LENGTH (LENGTH OF WS-ERR-DATA)
                    ERASE
```

```
           END-EXEC.
*
           GO TO  0000-RETURN.
*
         EJECT
*-------------------------------------------------------------*
       9900-ERR-DISPLAY.
*-------------------------------------------------------------*
*--ERROR IN "MQ" VERB
*
           MOVE WS-CCODE-VALUE TO  WS-ERR-DISPLAY-CCODE.
           MOVE WS-RCODE-VALUE TO  WS-ERR-DISPLAY-RCODE.
*
           EXEC CICS  SEND
                      FROM   (WS-ERR-DISPLAY)
                      LENGTH (LENGTH OF WS-ERR-DISPLAY)
                      ERASE
           END-EXEC.
*
           GO TO  0000-RETURN.
*
```

Sample program TTPTST2.Z

This program is a test facility for sending/receiving messages. It can be invoked either by terminal input or passed data (triggered by CICS “START”) format as that for `ttptst1`.

The difference from ttptst1 is the usage of `MQCONN, MQOPEN, MQCLOSE` and `MQDISC`. Regardless of the number of messages, there are only one connection and disconnection to the Queue Manager and one open and close of the processing queue.

```
*/INCLUDE COPYRSAP
*****************************************************************
* Licensed Materials - Property of IBM                          *
*                                                               *
* 5787-ECX                                                      *
* (C) Copyright IBM Corp. 1993, 1996                            *
*                                                               *
* US Government Users Restricted Rights - Use, duplication or   *
* disclosure restricted by GSA ADP Schedule Contract with IBM   *
* Corp.                                                         *
*****************************************************************
       IDENTIFICATION DIVISION.
       PROGRAM-ID.    TTPTST2.
       AUTHOR.        IBM.

       DATE-WRITTEN.  12/15/92.
       DATE-COMPILED.
*LAST-MODIFIED.  3/21/96.

*****************************************************************
*---------------------------------------------------------------*
*               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            *
*                                                               *
*                 IBM MQI                                       *
*                                                               *
*---------------------------------------------------------------*
* 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 --                                        *
*                                                               *
* 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(80) VALUE
            'Licensed Materials - Property of IBM'.
            05 FILLER                  PIC X(80) VALUE SPACES.
            05 FILLER                  PIC X(80) VALUE
            '5787-ECX  '.
            05 FILLER                  PIC X(80) VALUE SPACES.
            05 FILLER                  PIC X(80) VALUE
            '(C) Copyright IBM Corp. 1993, 1996   All Rights
                 Reserved'.
            05 FILLER                  PIC X(80) VALUE SPACES.
            05 FILLER                  PIC X(80) VALUE
            'US Government Users Restricted Rights - Use,
                 duplication '.
            05 FILLER                  PIC X(80) VALUE
           'or disclosure restricted by GSA ADP Schedule Contract
                 '.
            05 FILLER                  PIC X(80) VALUE
            'with IBM Corp.'.

*---------------------------------------------------------------*

       01  FILLER                        PIC X(40) VALUE
                    'TTPTST2 WORKING STORAGE STARTS HERE ==>'.

       01  WS-VERSION.
           05  FILLER                   PIC X(30)      VALUE
           'TTPTST2 VERSION 1.4'.

       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.

          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.
*/INCLUDE COPYROCO
*-----------------------------------------------------------------*
* - 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.
*/INCLUDE COPYROCO
*-----------------------------------------------------------------*
* 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-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  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  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.

*-------------------------------------------------------------*

       EJECT
*-------------------------------------------------------------*
*-------------------------------------------------------------*
      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  FILLER             PIC  X(5)  VALUE 'TST2 '.
                  15  WS-DATA-FUNCTION    PIC  XXXX  VALUE 'PUT'.
                   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'.

                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.
           EJECT
*-------------------------------------------------------------*
       01  WS-PASSED-INFO.

*    COPY   TTITST2.
*COPY COPYRSAP
*-------------------------------------------------------------*
* - 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
*---------------------------------------------------------------*
       01  WS-NEED-REPLY.
           05  FILLER                   PIC  X(80) VALUE
           'Please enter REPLY QUEUE name with trailing blanks or
                ErsEOF
      -    ' (e.g. 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
                QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ
      -    '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
           '                      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
           '  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 15 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.
           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).

*---------------------------------------------------------------*
       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  ='.
            1O  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.
*    COPY CMQV.
*/INCLUDE CMQV
*/INCLUDE  COPYR
*****************************************************************
**                                                             **
**  FILE NAME:        CMQV                                     **
**                                                             **
**  DESCRIPTIVE NAME: COBOL copy file for MQI constants        **
**                                                             **
**  VERSION 1.4.0                                              **
**                                                             **
**  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-SYNCPOINT              PIC S9(9) BINARY  VALUE 2.
         10 MQGMO-NO-SYNCPOINT           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-INTEGER-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-INTEGER-UNDEFINED PIC S9(9) BINARY  VALUE 0.
       10 MQENC-INTEGER-NORMAL    PIC S9(9) BINARY  VALUE 1.
       10 MQENC-INTEGER-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.

**   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-SYNCPOINT            PIC S9(9) BINARY  VALUE 2.
       10 MQPMO-NO-SYNCPOINT         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-SYNCPOINT-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
*-------------------------------------------------------------*
*    COPY MQICENV.
*/INCLUDE COPYROCO
*-------------------------------------------------------------*
* - 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-EZBRIDGE    VALUE 'EZB '.
                88  ENV-PRODUCT-MQM         VALUE 'MQM '.

            05 ENV-PRODUCT-RUNTIME        PIC  X(4) VALUE 'BOTH'.
                88  ENV-PRODUCT-RT-EZBRIDGE VALUE 'EZB '.
                88  ENV-PRODUCT-RT-MQM      VALUE 'MQM '.
                88  ENV-PRODUCT-RT-BOTH     VALUE 'BOTH'.

             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-MMDDYY                    VALUE 01.
               88 ENV-DATE-YYMMDD                    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 'MQO2'.
                 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
                'MQPMTP'.
                10 ENV-MT-CONFIG-PROGRAM    PIC  X(8) VALUE
                'MQPMCFG'.
                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
                '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
                'MQPMDEL'.
                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-AIPO         PIC  X(8) VALUE
                'MQPAIPO '.
                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
      'MQMMTP'.
      10 ENV-MT-CONFIG-MAPSCREEN  PIC  X(8) VALUE
      'MQMMCFG'.
      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-MONQ-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 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
    'MQ900000: 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.
*    COPY  CMQODV.
*/INCLUDE CMQODV
*/INCLUDE  COPYR
************************************************************
**                                                        **
**  FILE NAME:        CMQODV                              **
**                                                        **
**  DESCRIPTIVE NAME: COBOL copy file for MQOD structure  **
**                                                        **
**  VERSION 1.4.0                                       **
**                                                        **
**  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-SECTOR-COUNT.
           05  WS-SECTOR-COUNT            PIC S9(8) COMP.
       01  MQI-SECTOR.
           05  WS-SECTOR                  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.
*/INCCLUDE CMQMDV
*/INCLUDE  COPYR
************************************************************
**                                                        **
**  FILE NAME:        CMQMDV                              **
**                                                        **
**  DESCRIPTIVE NAME: COBOL copy file for MQMD structure  **
**                                                        **
**  VERSION 1.4.0                                       **
**                                                        **
**  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.
*/INCLUDE CMQPMOV
*/INCLUDE  COPYR
************************************************************
**                                                        **
**  FILE NAME:        CMQPMOV                             **
**                                                        **
**  DESCRIPTIVE NAME: COBOL copy file for MQPMO structure **
**                                                        **
**  VERSION 1.4.0                                       **
**                                                        **
**  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-KNOWNDESTCOUNT    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.
*    COPY   CMQGMOV.
*/INCLUDE CMQGMOV
*/INCLUDE  COPYR
************************************************************
**                                                        **
**  FILE NAME:        CMQGMOV                             **
**                                                        **
**  DESCRIPTIVE NAME: COBOL copy file for MQGMO structure **
**                                                        **
**  VERSION 1.4.0                                       **
```

```
**                                                               **
**  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-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.

       01  WS-ORIGINAL-BUFFER-AREA.
           05  FILLER                     PIC  X(200) VALUE
           'THIS IS A MESSAGE TEXT'.

*-----------------------------------------------------------------*
        EJECT
*-----------------------------------------------------------------*
       LINKAGE SECTION.
*-----------------------------------------------------------------*

       01  LK-DATA.
           05  FILLER                   PIC X(1000).
        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

           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 'TTPTST2'     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
           EXEC  CICS RECEIVE
                           SET(ADDRESS OF LK-DATA)
                        LENGTH(WS-DATA-LENGTH)
                 END-EXEC.

*--CHECK WHAT WE'RE DOING
*-- --COMMAND IS  "TST2 GET 01 QUEUENAME"
           IF (WS-DATA-LENGTH < LENGTH OF WS-DATA-WITH-FUNCTION)
           OR (WS-DATA-LENGTH > LENGTH OF WS-DATA-ALL)
            THEN
                PERFORM  1100-SEND-HELP
                    GO TO   0000-RETURN.

*--DO VARIABLE MOVE
           CALL 'MQPMOVE'  USING  WS-DATA-WITH-QUEUE
                                  LK-DATA
                                  WS-DATA-LENGTH.

*
           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.

*--DO VARIABLE MOVE
           CALL 'MQPMOVE'  USING  WS-REPLY-Q
                                  LK-DATA
                                  WS-DATA-LENGTH.


*
*-------------------------------------------------------------*
        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-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  SPACES    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 MQOO-BROWSE       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-BROWSE-FIRST   TO  MQGMO-OPTIONS
             ADD  MQGMO-ACCEPT-TRUNCATED-MSG
                                  TO  MQGMO-OPTIONS
             MOVE  SPACES    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
*-- --CHECK RC
             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
             END-IF
*

*--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  SPACES    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 MQOO-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 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

*--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  SPACES       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  SPACES    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
             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
*-------------------------------------------------------------*
       7000-SEND-TOTALS.
*-------------------------------------------------------------*
*--GET DURACTION 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.
```

```
                                                                    9999-ABEND-USER-CODE.
        EJECT
*-------------------------------------------------------------*
       9900-ERR-DISPLAY.                                            GO TO  0000-RETURN.
*-------------------------------------------------------------*
*--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 EIBRSRCE         TO ERR-DEBUG-EIBRSRCE.

           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 ******
```

Sample program TTPTST3.Z

This program is a test facility for putting/getting messages by starting a transaction TST2 (program id TTPTST2). It can be invoked either by terminal input or passed data (triggered by CICS “START”).

The terminal input is “TST3” and the response is a screen requesting more input:

```
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 38. Test System Programs 3 - start

On this screen the fields are:

Async TASK Information

Number of tasks: The number of asynchronized tasks (TST2 transactions).

Message Processing Information

Number of messages: The number of messages to be sent/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 PUT function only, to specify the size of the message. If the PUT timestamp option is selected, the message will be 16 characters greater than the PUT message size.

PUT message: The content of the message.

PUT TimeStamp: For PUT function only, to put time stamp in the message, format as YYMMDDHHMMSS. If “Y” is specified, the actual message size will be 16 characters greater than the size specified in the PUT message size.

```
*/INCLUDE COPYRSAP
*****************************************************************
* Licensed Materials - Property of IBM                          *
*                                                               *
* 5787-ECX                                                      *
* (C) Copyright IBM Corp. 1993, 1996                            *
*                                                               *
* US Government Users Restricted Rights - Use, duplication or   *
* disclosure restricted by GSA ADP Schedule Contract with IBM   *
* Corp.                                                         *
*****************************************************************
       IDENTIFICATION DIVISION.
       PROGRAM-ID.    TTPTST3.
       AUTHOR.        IBM.

       DATE-WRITTEN.   2/23/93.
       DATE-COMPILED.
      *LAST-MODIFIED.  9/ 1/95.

*---------------------------------------------------------------*
*---------------------------------------------------------------*
*                                                               *
*---------------------------------------------------------------*
*                                                               *
*  TEST PROGRAMS TO START ASYNC TASKS                           *
*                                                               *
*              IBM MQI SYSTEM                                   *
*                                                               *
*---------------------------------------------------------------*
* 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            *
*                                                              *
* 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(80) VALUE
            'Licensed Materials - Property of IBM'.
            05 FILLER                 PIC X(80) VALUE SPACES.
            05 FILLER                 PIC X(80) VALUE
            '5787-ECX  '.
            05 FILLER                 PIC X(80) VALUE SPACES.
            05 FILLER                 PIC X(80) VALUE
            '(C) Copyright IBM Corp. 1993, 1996   All Rights
                 Reserved'.
            05 FILLER                 PIC X(80) VALUE SPACES.
            05 FILLER                 PIC X(80) VALUE
            'US Government Users Restricted Rights - Use,
                 duplication '.
            05 FILLER                 PIC X(80) VALUE
           'or disclosure restricted by GSA ADP Schedule Contract
                '.
            05 FILLER                 PIC X(80) VALUE
            'with IBM Corp.'.

*--------------------------------------------------------------*

       01  WS-VERSION.
           05  FILLER                 PIC X(30)      VALUE
           'TTPTST3 VERSION 1.4'.

       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 '/'.
               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-TRAN-ID                PIC  X(4)  VALUE SPACES.
           05  WS-EDIT-ERR-FLAG          PIC  X(1) VALUE 'N'.
            88 WS-EDIT-ERR                       VALUE 'Y'.

           05  WS-RECORD-FLAG           PIC  X      VALUE SPACES.
            88 WS-RECORD-FOUND          VALUE 'T'.
            88 WS-RECORD-NOT-FOUND      VALUE 'F'.

           05  WS-ERROR-MESSAGE         PIC  X(79)  VALUE SPACES.
           05  WS-ERR-COUNT            PIC S9(4) COMP VALUE ZEROS.
           05  WS-ERR-MAX               PIC S9(4) COMP VALUE +20.
           05  WS-ERR-MESSAGE  VALUE SPACES.
            10  WS-ERR-MSG             PIC  X(79) OCCURS 20 TIMES.

*--------------------------------------------------------------*
          EJECT
*--------------------------------------------------------------*
           COPY DFHAID.
          EJECT
*--------------------------------------------------------------*
           COPY DFHBMSCA.

          EJECT
*--------------------------------------------------------------*
* BMS MAP
*--------------------------------------------------------------*
           COPY TTMTST3.
           EJECT
*--------------------------------------------------------------*
      * TST2 COMMAREA
*--------------------------------------------------------------*
       01  WS-TST2-COMMAREA.
*    COPY TTITST2.
*COPY COPYRSAP
*--------------------------------------------------------------*
* - 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
```

```
*-------------------------------------------------------------*
* ENVIRNOMENT VALUES
*-------------------------------------------------------------*
      01  FILLER.
*    COPY MQICENV.
*/INCLUDE COPYROCO
*-------------------------------------------------------------*
* - 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-EZBRIDGE    VALUE 'EZB '.
              88  ENV-PRODUCT-MQM         VALUE 'MQM '.

           05 ENV-PRODUCT-RUNTIME        PIC  X(4) VALUE 'BOTH'.
              88  ENV-PRODUCT-RT-EZBRIDGE VALUE 'EZB '.
              88  ENV-PRODUCT-RT-MQM      VALUE 'MQM '.
              88  ENV-PRODUCT-RT-BOTH     VALUE 'BOTH'.

            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-MMDDYY                  VALUE 01.
              88 ENV-DATE-YYMMDD                  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
                '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.


          03 ENV-DATA-FOR-PROGRAMS.

             05 ENV-MASTER-TERMINAL-PROGRAMS.
                10 ENV-MT-MASTER-PROGRAM    PIC  X(8) VALUE
                'MQPMTP'.
                10 ENV-MT-CONFIG-PROGRAM    PIC  X(8) VALUE
                'MQPMCFG'.
                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
                '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
                'MQPMDEL'.
                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
      'MQMMTP'.
      10 ENV-MT-CONFIG-MAPSCREEN  PIC  X(8) VALUE
      'MQMMCFG'.
      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-MONQ-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 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
   'MQ900000: MQSERIES VSE ENVIRONMENT not initialized.'.
      10 FILLER                  PIC  X(80) VALUE SPACES.

*-----------------------------------------------------------*
* - END   -      ***  COPYBOOK: MQICENV   ***         - END   - *
*-----------------------------------------------------------*
          EJECT
*-----------------------------------------------------------*
* FINDQ COMMAREA
*-----------------------------------------------------------*
       01  WS-FINDQ.
*    COPY MQIFINDQ.
*/INCLUDE COPYROCO
*-----------------------------------------------------------*
* - 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-TERMID           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.
*/INCLUDE COPYROCO
*-------------------------------------------------------------*
*-------------------------------------------------------------*
* 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  '.

           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.
*/INCLUDE COPYROCO
*---------------------------------------------------------------*
* - 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).
                88 RECORD-TYPE-IS-ENV             VALUE 'ENV'.
              05 ENV-RECORD-VERSION           PIC  9(4).
              05 ENV-RECORD-TYPE              PIC  X(4).
                88 ENV-TYPE-SYSTEM                VALUE 'SYS'.
                88 ENV-TYPE-TRANACTION            VALUE 'TRAN'.
                88 ENV-TYPE-PROGRAM               VALUE 'PROG'.
                88 ENV-TYPE-MAPS                  VALUE 'MAPS'.
                88 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(8).
              05 ENV-LAST-USERID              PIC  X(3).
              05 FILLER                       PIC  X.

           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(8).
              05 ENV-ADDED-USERID             PIC  X(3).
              05 FILLER                       PIC  X.

           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).
                88 RECORD-TYPE-IS-SYS             VALUE 'SYS'.
              05 SYS-RECORD-SYSTEM-NUMBER     PIC  9(4).
              05 SYS-RECORD-TYPE              PIC  X(4).
                88 SYS-TYPE-SYS                   VALUE 'SYS'.
                88 SYS-TYPE-QUE-MAX               VALUE 'QUEM'.
                88 SYS-TYPE-QUE-DEFAULT           VALUE 'QUED'.
                88 SYS-TYPE-COM-MAX               VALUE 'COMM'.
                88 SYS-TYPE-COM-DEFAULT           VALUE 'COMD'.
                88 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(8).
              05 SYS-LAST-USERID              PIC  X(3).
              05 FILLER                       PIC  X.

           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(8).
              05 SYS-ADDED-USERID             PIC  X(3).
              05 FILLER                       PIC  X.

           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).
                88 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(8).
              05 QDR-LAST-USERID              PIC  X(3).
              05 FILLER                       PIC  X.

           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(8).
              05 QDR-ADDED-USERID             PIC  X(3).
              05 FILLER                       PIC  X.

           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-TERMID                PIC  X(8).
              05 COM-LAST-USERID                PIC  X(3).
              05 FILLER                         PIC  X.

           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-TERMID               PIC  X(8).
              05 COM-ADDED-USERID               PIC  X(3).
              05 FILLER                         PIC  X.


           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-TERMID               PIC  X(8).
              05 TEXT-LAST-USERID               PIC  X(3).
              05 FILLER                         PIC  X.


           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-TERMID              PIC  X(8).
              05 TEXT-ADDED-USERID              PIC  X(3).
              05 FILLER                         PIC  X.


           03 TEXT-DATA-AREA.
              05 FILLER                         PIC  X(1892).

*-------------------------------------------------------------*
* - END -       ***  COPYBOOK: MQICONFG  ***           - END - *
*-------------------------------------------------------------*


           EJECT
*-------------------------------------------------------------*
* ERROR HANDLEING
*-------------------------------------------------------------*
       01  WS-ERR.
*    COPY        MQIERR.
*/INCLUDE COPYROCO
*-------------------------------------------------------------*
* - 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.
*/INCLUDE COPYROCO
*---------------------------------------------------------------*
* 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-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  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  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.

*-------------------------------------------------------------*

*    COPY TTETST3.
*COPY COPYROCO
*-------------------------------------------------------------*
* 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 NME 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. '.

*-------------------------------------------------------------*

           EJECT
*-------------------------------------------------------------*
       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.
*
          EXEC  CICS FORMATTIME
                     ABSTIME(WS-ABSTIME)
                     YYMMDD (WS-DATE-YYMMDD)
                END-EXEC.

*-- --SET CENTURY
          IF WS-DATE-YY >  50
           THEN
                 MOVE  19                TO  WS-DATE-CC
           ELSE
                 MOVE  20                TO  WS-DATE-CC.

*--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
                                        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
                       MOVE DFHUNIMD TO LTNUMA
                       MOVE 'Y'      TO WS-EDIT-ERR-FLAG
                       MOVE MSG-ERR-MAX-TASK-VALUE
                                     TO WS-ERROR-MESSAGE
                       PERFORM  8000-MOVE-ERR-MESSAGE
                     ELSE
                       MOVE WS-NUM TO WS-SS-STARTS
              ELSE
                    MOVE -1       TO LTNUML
                    MOVE DFHUNIMD TO LTNUMA
                    MOVE 'Y'      TO WS-EDIT-ERR-FLAG
                    MOVE MSG-ERR-MAX-TASK-VALUE
                                  TO WS-ERROR-MESSAGE
                    PERFORM  8000-MOVE-ERR-MESSAGE.

*--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)
                         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 (MAINO)
               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 WS-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 (MAINO)
                            ERASE CURSOR
                   END-EXEC
            ELSE
              MOVE  WS-ERROR-MESSAGE  TO  LERRO
              EXEC CICS SEND MAP (MTP-MAP-VALUE)
                          MAPSET('TTMTST3')
                           FROM (MAINO)
                            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.
*-------------------------------------------------------------*
* 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
*-------------------------------------------------------------*
       9000-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 LTNUML.
           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.
*-------------------------------------------------------------*
* 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 EIBRSRCE         TO ERR-DEBUG-EIBRSRCE.
```

```
           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.


*--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 COPYRASP.
*****************************************************************
* COPYBOOK COPYRSAP
* Licensed Materials - Property of IBM                          *
*                                                               *
* 5787-ECX                                                      *
* (C) Copyright IBM Corp. 1993, 1994                            *
*                                                               *
* US Government Users Restricted Rights - Use, duplication or   *
* disclosure restricted by GSA ADP Schedule Contract with IBM   *
* Corp.                                                         *
*****************************************************************
       IDENTIFICATION DIVISION.
       PROGRAM-ID.    MQPECHO.
       AUTHOR.        IBM


       DATE-WRITTEN.   9/ 1/92.
       DATE-COMPILED.
      *LAST-MODIFIED.  9/ 1/95.

*****************************************************************
*---------------------------------------------------------------*
*---------------------------------------------------------------*
*               T E S T     ECHO                              *
*                                                               *
*          A P P L I C A T I O N   I N T E R F A C E            *
*                                                               *
*                MQSeries for VSE/ESA                       *
*                                                               *
*---------------------------------------------------------------*
* MQPECHO - IBM APPLICATION TEST PROGRAM                        *
*                                                               *
* PREREQUISITE:                                               *
*           1. SENDING QUEUE, A LOCAL QUEUE NAMED XXX,        *
*              MUST BE DEFINED WITH                             *
*                TRIGGER ENABLE:  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 OUT  99 XXX.                      *
*              C. 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       *
*                MORE MESSAGE.                               *
*             3. ECHO READ MESSAGES INTO IBM.REPLY.QUEUE1  *
*                                                                *
* 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(80) VALUE
            'Licensed Materials - Property of IBM'.
            05 FILLER                  PIC X(80) VALUE SPACES.
            05 FILLER                  PIC X(80) VALUE
            '5787-ECX  '.
            05 FILLER                  PIC X(80) VALUE SPACES.
            05 FILLER                  PIC X(80) VALUE
            '(C) Copyright IBM Corp. 1993, 1996   All Rights
                Reserved'.
            05 FILLER                  PIC X(80) VALUE SPACES.
            05 FILLER                  PIC X(80) VALUE
            'US Government Users Restricted Rights - Use,
                duplication '.
            05 FILLER                  PIC X(80) VALUE
          'or disclosure restricted by GSA ADP Schedule Contract
                '.
            05 FILLER                  PIC X(80) VALUE
            'with IBM Corp.'.

*---------------------------------------------------------------*

*---------------------------------------------------------------*
       01 FILLER    PIC X(40) VALUE
            'MQPECHO VERSION 1.4.0.'.

*---------------------------------------------------------------*

       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-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 '/'.
           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'.

*-------------------------------------------------------------*
      EJECT
*-------------------------------------------------------------*
     * 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.

*
*-------------------------------------------------------------*
      EJECT
*-------------------------------------------------------------*
* 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-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-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  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  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.

*-------------------------------------------------------------*
       EJECT
*-------------------------------------------------------------*
* 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 '.
               88  ENV-PRODUCT-EZBRIDGE    VALUE 'EZB '.
               88  ENV-PRODUCT-MQM         VALUE 'MQM '.

            05 ENV-PRODUCT-RUNTIME        PIC  X(4) VALUE 'BOTH'.
               88  ENV-PRODUCT-RT-EZBRIDGE VALUE 'EZB '.
               88  ENV-PRODUCT-RT-MQM      VALUE 'MQM '.
               88  ENV-PRODUCT-RT-BOTH     VALUE 'BOTH'.

            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-MMDDYY                  VALUE 01.
              88 ENV-DATE-YYMMDD                  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 'MQO2'.
      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
      'MQPMTP'.
      10 ENV-MT-CONFIG-PROGRAM    PIC  X(8) VALUE
      'MQPMCFG'.
      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
      '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
      'MQPMDEL'.
      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
      'MQMMTP'.
      10 ENV-MT-CONFIG-MAPSCREEN  PIC  X(8) VALUE
      'MQMMCFG'.
      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-MONQ-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 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
            'MQ900000: MQSERIES VSE ENVIRONMENT not initialized.'.
                10 FILLER                  PIC  X(80) VALUE SPACES.

*-----------------------------------------------------------------*
* - END   -      ***  COPYBOOK: MQICENV   ***          - END   - *
*-----------------------------------------------------------------*
       EJECT
*-----------------------------------------------------------------*
* 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).

*-----------------------------------------------------------------*
       EJECT
*-----------------------------------------------------------------*
       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-SYNCPOINT              PIC S9(9) BINARY  VALUE 2.
        10 MQGMO-NO-SYNCPOINT           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-INTEGER-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-INTEGER-UNDEFINED PIC S9(9) BINARY  VALUE 0.
       10 MQENC-INTEGER-NORMAL    PIC S9(9) BINARY  VALUE 1.
       10 MQENC-INTEGER-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.

**   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-SYNCPOINT            PIC S9(9) BINARY  VALUE 2.
       10 MQPMO-NO-SYNCPOINT         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-SYNCPOINT-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
*----------------------------------------------------------------*
* 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 - *
*-----------------------------------------------------------------*
         EJECT
*-----------------------------------------------------------------*
*     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 'O'.
               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 'O'.
              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 '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.
             EJECT

*--INQ
       01  MQI-SECTOR-COUNT.
           05  WS-SECTOR-COUNT            PIC S9(8) COMP.
       01  MQI-SECTOR.
           05  WS-SECTOR                  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                               **
*                                                         **
**  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.
**    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-KNOWNDESTCOUNT   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.
*    COPY    CMQGMOV.
****************************************************************
**                                                            **
**  FILE NAME:        CMQGMOV                                 **
**                                                            **
**  DESCRIPTIVE NAME: COBOL copy file for MQGMO structure     **
**                                                            **
**  VERSION 1.3.0                                           **
**                                                            **
**  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.

*--------------------------------------------------------------*
* 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).
        EJECT
*--------------------------------------------------------------*
       LINKAGE SECTION.
*--------------------------------------------------------------*
       01  DFHCOMMAREA.
           05  FILLER                   PIC X(1000).

*--------------------------------------------------------------*
        EJECT
*--------------------------------------------------------------*
```

```
       PROCEDURE DIVISION.
*-------------------------------------------------------------*

       0000-MAIN-LINE.

*--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.
        EJECT
*-------------------------------------------------------------*
       1000-INITIALIZE.
*-------------------------------------------------------------*
* PURPOSE: SETUP DATA AREAS
*-------------------------------------------------------------*

*--GET ENVIRONMENT INFO
           PERFORM 1015-GET-ENVRIR-RECORD
              THRU 1015-GET-ENVRIR-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.
        EJECT
*-------------------------------------------------------------*
       1015-GET-ENVRIR-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-ENVRIR-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.
*
           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.

*-------------------------------------------------------------*
        EJECT
*-------------------------------------------------------------*
       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.
*
*-------------------------------------------------------------*
       EJECT
*-------------------------------------------------------------*
      3100-GET-OPEN.
*-------------------------------------------------------------*
* PURPOSE: OPEN
*-------------------------------------------------------------*
*--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 MQI-PROC-LOCAL-QUEUE-NAME
                                 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-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.

*
*-------------------------------------------------------------*
       EJECT
*-------------------------------------------------------------*
      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  SPACES    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.


*--SYNCPOINT
          EXEC CICS SYNCPOINT
          END-EXEC.
*
*
*-------------------------------------------------------------*
        EJECT
*-------------------------------------------------------------*
      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.


*-------------------------------------------------------------*
       EJECT
*-------------------------------------------------------------*
      4000-PUT1-MESSAGES.
*-------------------------------------------------------------*
* PURPOSE: PUT1
*-------------------------------------------------------------*
*--MQPUT1   QUEUE TO QM
          MOVE  'PUT1'    TO   WS-FUNCTION.
          MOVE MQOO-OUTPUT    TO     WS-Q-OPEN-OPTIONS-VALUE.
          IF MQMD-REPLYTOQMGR    EQUAL SPACES OR LOW-VALUES
                MOVE SPACES   TO   MQOD-OBJECTQMGRNAME
           ELSE
                MOVE MQMD-REPLYTOQMGR
                              TO   MQOD-OBJECTQMGRNAME
          END-IF.
*
*--IF NOT REPLY QUEUE - SET DEFAULT
          IF MQMD-REPLYTOQ     EQUAL SPACES OR LOW-VALUES
                MOVE WS-R-Q-NAME
                              TO   MQOD-OBJECTNAME
           ELSE
                MOVE MQMD-REPLYTOQ
                              TO   MQOD-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.
        EJECT
*-------------------------------------------------------------*
       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.
*

*-------------------------------------------------------------*
         EJECT
*-------------------------------------------------------------*
       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.
*
            EJECT
*-------------------------------------------------------------*
       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.

         EJECT
*-------------------------------------------------------------*
* ERROR PROCESSING
*-------------------------------------------------------------*
*    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 EIBRSRCE         TO ERR-DEBUG-EIBRSRCE.

           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.

*--ADDED CODE FOR ABEND CONDITION
*--RETURN
           GO TO  0000-RETURN.

       9900-NO-ENVIR-SETUP.
           GO TO  0000-RETURN.
```

Appendix E. COBOL copybooks

CMQDLHV.C

```
*****************************************************************
 ** FILE NAME: CMQDLHV                                          **
 **                                                             **
 **                                                             **
 ** DESCRIPTIVE NAME: COBOL copy file for MQDLH structure       **
 **                                                             **
 ** FUNCTION: This file declares the MQDLH structure,           **
 ** which forms part of the IBM Message                         **
 ** Queue Interface (MQI).                                      **
*****************************************************************
************************************************************
 ** Licensed Materials - Property of IBM                       **
 **                                                            **
 ** This Module is Restricted Materials of IBM                 **
 ** 5787-ECX                                                   **
 ** © Copyright IBM Corp. 1993, 1996                           **
 **                                                            **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                         **
************************************************************
**  MQDLH structure
   10 MQDLH.
**  Structure identifier
   15 MQDLH-STRUCID
                PIC X(4) 'DLH '.
**  Structure version number
   15 MQDLH-VERSION
                PIC S9(9) BINARY VALUE ZERO.
**  Reason message arrived on dead-letter queue
   15 MQDLH-REASON
                PIC S9(9) BINARY VALUE ZERO.
**  Name of original destination queue
   15 MQDLH-DESTQNAME
                PIC X(48) VALUE SPACES.
** Name of original destination queue manager
   15 MQDLH-DESTQMGRNAME
                PIC X(48) VALUE SPACES.
**  Original data encoding
   15 MQDLH-ENCODING
                PIC S9(9) BINARY VALUE ZERO.
**  Original coded character set identifier
   15 MQDLH-CODEDCHARSETID
                PIC S9(9) BINARY VALUE ZERO.
**  Original format name
   15 MQDLH-FORMAT
                PIC X(8) VALUE SPACES.
**  Type of application that put message on dead-letter queue
   15 MQDLH-PUTAPPLTYPE
                PIC S9(9) BINARY VALUE ZERO.
**  Name of application that put message on dead-letter queue
   15 MQDLH-PUTAPPLNAME
                PIC X(28) VALUE SPACES.
**  Date when message was put on dead-letter queue
   15 MQDLH-PUTDATE
                PIC X(8) VALUE SPACES.
**  Time when message was put on dead-letter queue
   15 MQDLH-PUTTIME
                PIC X(8) VALUE SPACES.
```

CMQGMOV.C

```
*****************************************************************
**                                                             **
** FILE NAME: CMQGMOV                                          **
**                                                             **
** DESCRIPTIVE NAME: COBOL copy file for MQGMO structure       **
**                                                             **
** FUNCTION: This file declares the MQGMO structure,           **
** which forms part of the IBM Message                         **
** Queue Interface (MQI).                                      **
**                                                             **
*****************************************************************

************************************************************
 ** Licensed Materials - Property of IBM                       **
 **                                                            **
 ** This Module is Restricted Materials of IBM                 **
 ** 5787-ECX                                                   **
 ** © Copyright IBM Corp. 1993, 1996                           **
 **                                                            **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                         **
************************************************************

 ** MQGMO structure
   10 MQGMO.
 ** Structure identifier
   15 MQGMO-STRUCID PIC X(4) VALUE 'GMO '.
 ** Structure version number
   15 MQGMO-VERSION PIC S9(8) BINARY VALUE 1.
 ** Options
   15 MQGMO-OPTIONS PIC S9(8) BINARY VALUE 0.
 ** Wait interval
   15 MQGMO-WAITINTERVAL PIC S9(8) BINARY VALUE 0.
 ** Signal
   15 MQGMO-SIGNAL1 PIC S9(8) BINARY VALUE 0.
 ** Reserved
   15 MQGMO-SIGNAL2 PIC S9(8) BINARY VALUE 0.
 ** Resolved name of destination queue
   15 MQGMO-RESOLVEDQNAME PIC X(48) VALUE SPACES.
```

CMQMDV.C

```
************************************************************
 **                                                        **
 ** FILE NAME: CMQMDV                                      **
 **                                                        **
 ** DESCRIPTIVE NAME: COBOL copy file for MQMD structure    **
 **                                                        **
 ** FUNCTION: This file declares the MQMD structure,       **
 ** which forms part of the IBM Message                    **
 ** Queue Interface (MQI).                                 **
 **                                                        **
************************************************************

************************************************************
 ** Licensed Materials - Property of IBM                   **
 **                                                        **
 ** This Module is Restricted Materials of IBM             **
 ** 5787-ECX                                               **
 ** © Copyright IBM Corp. 1993, 1996                       **
 **                                                        **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                     **
************************************************************
 ** MQMD structure
   10 MQMD.
 ** Structure identifier
   15 MQMD-STRUCID PIC X(4) VALUE 'MD '.
   88 MQMD-STRUCID-VALUE VALUE 'MD '.
 ** Structure version number
   15 MQMD-VERSION PIC S9(8) BINARY VALUE 1.
   88 MQMD-VERSION-VALUE VALUE 1.
 ** Reserved
   15 MQMD-REPORT PIC S9(8) BINARY VALUE 0.
   88 MQMD-REPORT-VALUE VALUE 0.
 ** Message type
   15 MQMD-MSGTYPE PIC S9(8) BINARY VALUE 8.
   88 MQMD-MSGTYPE-VALUE VALUE 8.
 ** Reserved
   15 MQMD-EXPIRY PIC S9(8) BINARY VALUE -1.
   88 MQMD-EXPIRY-VALUE VALUE -1.
 ** Feedback code
   15 MQMD-FEEDBACK PIC S9(8) BINARY VALUE 0.
   88 MQMD-FEEDBACK-VALUE VALUE 0.
 ** Data encoding
   15 MQMD-ENCODING PIC S9(8) BINARY VALUE 785.
   88 MQMD-ENCODING-VALUE VALUE 785.
 ** Coded character set identifier
   15 MQMD-CODEDCHARSETID PIC S9(8) BINARY VALUE 0.
   88 MQMD-CODEDCHARSETID-VALUE VALUE 0.
 ** Format name
   15 MQMD-FORMAT PIC X(8) VALUE SPACES.
 ** Reserved
   15 MQMD-PRIORITY PIC S9(8) BINARY VALUE 0.
   88 MQMD-PRIORITY-VALUE VALUE 0.
 ** Message persistence
   15 MQMD-PERSISTENCE PIC S9(8) BINARY VALUE 2.
   88 MQMD-PERSISTENCE-VALUE VALUE 2.
 ** Message identifier
   15 MQMD-MSGID PIC X(24) VALUE LOW-VALUES.
   88 MQMD-MSGID-VALUE VALUE LOW-VALUES.
 ** Correlation identifier
   15 MQMD-CORRELID PIC X(24) VALUE LOW-VALUES.
   88 MQMD-CORRELID-VALUE VALUE LOW-VALUES.
 ** Reserved
   15 MQMD-BACKOUTCOUNT PIC S9(8) BINARY VALUE 0.
   88 MQMD-BACKOUTCOUNT-VALUE 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.
   88 MQMD-ACCOUNTINGTOKEN-VALUE VALUE LOW-VALUES.
 ** Reserved
   15 MQMD-APPLIDENTITYDATA PIC X(32) VALUE SPACES.
 ** Reserved
   15 MQMD-PUTAPPLTYPE PIC S9(8) BINARY VALUE 0.
   88 MQMD-PUTAPPLTYPE-VALUE 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.
```

CMQODV.C

```
************************************************************
 **                                                        **
 ** FILE NAME: CMQODV                                      **
 **                                                        **
 ** DESCRIPTIVE NAME: COBOL copy file for MQOD structure   **
 **                                                        **
 ** FUNCTION: This file declares the MQOD structure,       **
 ** which forms part of the IBM Message                    **
 ** Queue Interface (MQI).                                 **
 **                                                        **
************************************************************
************************************************************
 ** Licensed Materials - Property of IBM                   **
 **                                                        **
 ** This Module is Restricted Materials of IBM             **
 ** 5787-ECX                                               **
 ** © Copyright IBM Corp. 1993, 1996                       **
 **                                                        **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                     **
************************************************************

 ** 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.
```

CMQPMOV.C

```
************************************************************
 **                                                        **
 ** FILE NAME: CMQPMOV                                     **
 **                                                        **
 ** DESCRIPTIVE NAME: COBOL copy file for MQPMO structure  **
 **                                                        **
 ** FUNCTION: This file declares the MQPMO structure,      **
 ** which forms part of the IBM Message                    **
 ** Queue Interface (MQI).                                 **
 **                                                        **
************************************************************
************************************************************
 ** Licensed Materials - Property of IBM                   **
 **                                                        **
 ** This Module is Restricted Materials of IBM             **
 ** 5787-ECX                                               **
 ** © Copyright IBM Corp. 1993, 1996                       **
 **                                                        **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                     **
************************************************************

 ** MQPMO structure
   10 MQPMO.
 ** Structure identifier
   15 MQPMO-STRUCID PIC X(4) VALUE 'PMO '.
 ** Structure version number
   15 MQPMO-VERSION PIC S9(8) BINARY VALUE 1.
 ** Reserved
   15 MQPMO-OPTIONS PIC S9(8) BINARY VALUE 0.
 ** Reserved
   15 MQPMO-TIMEOUT PIC S9(8) BINARY VALUE -1.
 ** Reserved
   15 MQPMO-CONTEXT PIC S9(8) BINARY VALUE 0.
 ** Reserved
   15 MQPMO-KNOWNDESTCOUNT PIC S9(8) BINARY VALUE 0.
 ** Reserved
   15 MQPMO-UNKNOWNDESTCOUNT PIC S9(8) BINARY VALUE 0.
 ** Reserved
   15 MQPMO-INVALIDDESTCOUNT PIC S9(8) 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.
```

CMQTMV.C

```
************************************************************
 **                                                        **
 ** FILE NAME: CMQTMV                                      **
 **                                                        **
 ** DESCRIPTIVE NAME: COBOL copy file for MQTM structure   **
 **                                                        **
 **                                                        **
 ** FUNCTION: This file declares the MQTM structure,       **
 ** which forms part of the IBM Message                    **
 ** Queue Interface (MQI).                                 **
 **                                                        **
************************************************************

************************************************************
 ** Licensed Materials - Property of IBM                   **
 **                                                        **
 ** This Module is Restricted Materials of IBM             **
 ** 5787-ECX                                               **
 ** © Copyright IBM Corp. 1993, 1996                       **
 **                                                        **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                     **
************************************************************
**  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).
```

CMQV.C

```
************************************************************
** FILE NAME: CMQV                                        **
**                                                        **
** DESCRIPTIVE NAME: COBOL copy file for MQI constants    **
**                                                        **
** FUNCTION: This file declares the constants             **
** which form part of the IBM Message                     **
** Queue Interface (MQI).                                 **
**                                                        **
************************************************************
************************************************************
 ** Licensed Materials - Property of IBM                   **
 **                                                        **
 ** This Module is Restricted Materials of IBM             **
 ** 5787-ECX                                               **
 ** © Copyright IBM Corp. 1993, 1996                       **
 **                                                        **
 ** US Government Users Restricted Rights - Use, duplication or**
 ** disclosure restricted by GSA ADP Schedule Contract with IBM
                Corp.                                     **
************************************************************

************************************************************
** 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-SYNCPOINT
                PIC S9(9) BINARY VALUE 2.
   10 MQGMO-NO-SYNCPOINT
                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-INTEGER-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-INTEGER-UNDEFINED
                PIC S9(9) BINARY VALUE 0.
   10 MQENC-INTEGER-NORMAL
                PIC S9(9) BINARY VALUE 1.
   10 MQENC-INTEGER-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.

 ** 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-SYNCPOINT
                PIC S9(9) BINARY VALUE 2.
    10 MQPMO-NO-SYNCPOINT
                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.

** BINARYletion 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-SYNCPOINT-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.
```

Appendix F. Configuration worksheets

System list - worksheet

One list to be compiled for entire network, identifying all systems which will utilize MQSeries messaging and queuing. Each is assigned one *message_queue_manager* name.

System	Location	Hardware	Msg_Queue_Manager	Comments

Column 1 = System Name or identification (User specified terminology)

Column 2 = Location of system (City, Building, Floor, etc.)

Column 3 = Type of Hardware (Mainframe, LAN, AS/400, VAX, TANDEM, etc.)

Column 4 = Assigned Message_Queue_Manager name

Column 5 = Any other user comments

Application list - worksheet

One list to be compiled for entire network, identifying all applications which will utilize MQSeries messaging and queuing. Each is assigned one (or more) local *queue_name(s)* through which they will receive messages. Each is mapped to a host system from above list.

Application	Queue_Name	Hardware	Msg_Q_Mgr	Comments

Column 1 = Application Name or identification

Column 2 = Assigned local *Queue_Name*.

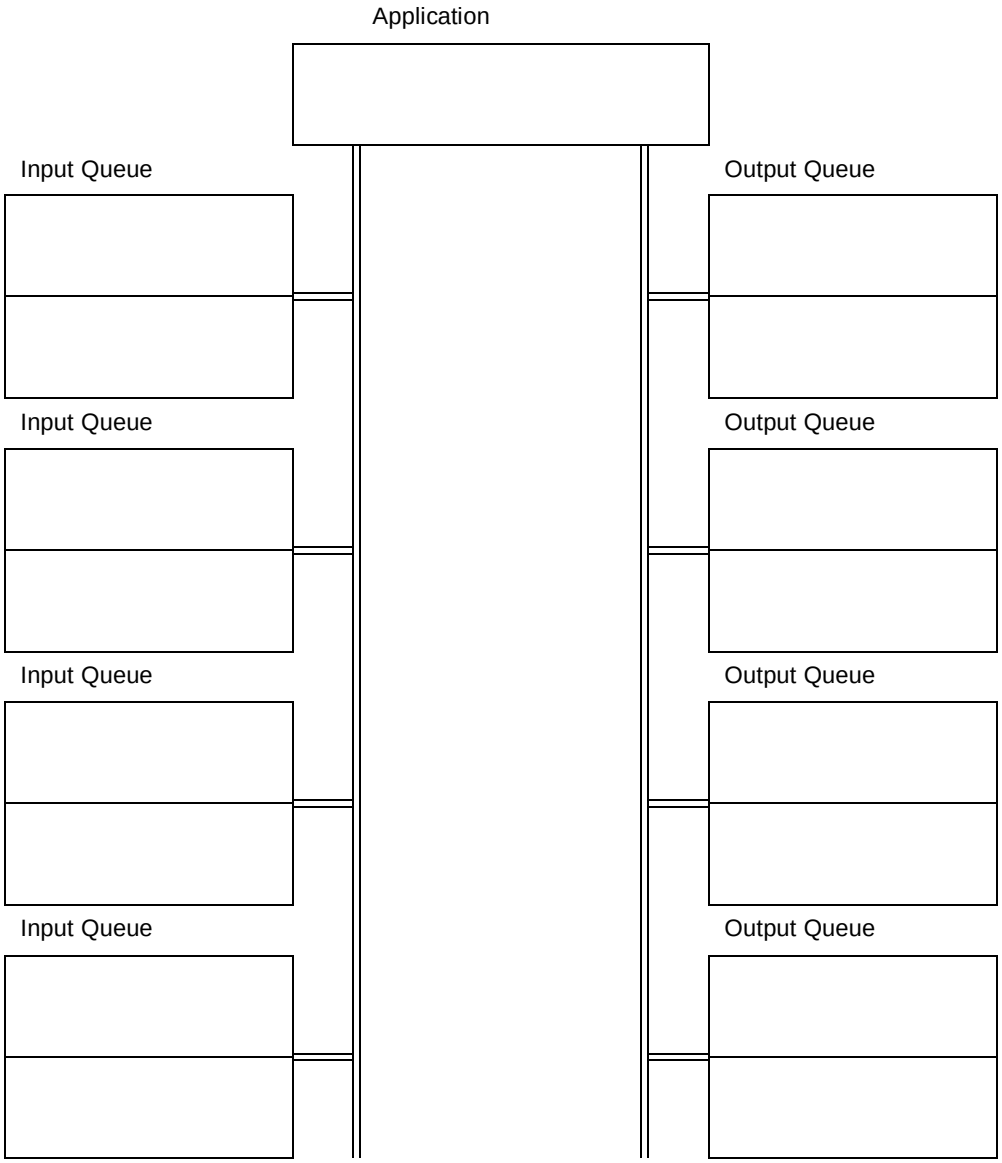
Column 3 = Type of Hardware (from SYSTEM LIST WORKSHEET)

Column 4 = Message_Queue_Manager name (from SYSTEM LIST WORKSHEET)

Column 5 = Any other user comments

Application look at queues - worksheet

One list to be compiled for **each application**, identifying all queues with which that application will interact. (This is primary input data to applications developers.)



Each queue box contains:

QUEUE_NAME

and

MESSAGE FORMAT (User Supplied Information)

System look at queues - worksheet

One list to be compiled for **each MQSeries System**. All applications on the system are identified, all queues required on the system are identified, all channels are identified. All data is derived from previous worksheets.

Local System			Remote System	
Application	**Queue_Name**	Channel	**Queue_Name**	**Queue_Manager_Name**
Input from Remote				
		<---		
		<---		
		<---		
		<---		
Output to Remote				
		--->		
		--->		
		--->		
		--->		
Local Messaging				
		None		
		None		
		None		
		None		
		None		
Passthru Cases *(this system is intermediate node in multi-hop routing)*				
		/---		
		\--->		
		/---		
		\--->		

Column 1 = Local Application Name or identification

Column 2 = Assigned local *Queue_Name*.

Column 3 = Channel (direction of message flow)

Column 4 = Remote system Queue_Name

Column 5 = Remote system Message_Queue_Manager name

Channel list - worksheet

One list to be compiled for entire network. All information in first 5 columns is derived from prior worksheets. All Channel names assigned (last column).

System	Q_Mgr_Name	Channel	System	Q_Mgr_Name	Channel_Name
		<---			
		--->			

Column 1 = System Name

Column 2 = Queue_Manager_Name for above system.

Column 3 = Channel (direction of message flow)

Column 4 = System Name of “other” system in connection

Column 5 = Queue_Manager_Name for above system

Column 6 = Assigned Channel Name

MQSeries System configuration (routing table) - worksheet

One list to be compiled for **each system**.

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
	Local			
	Local			
	Local			
	Local			
	Transmit			
	Transmit			
	Remote			
	Remote			
	Remote			

Entries are any of the following formats

Obj_Name	Type	Q_Name	QMgr_Name	Xmit_QName
Required	**Local**	n.a.	n.a.	n.a.
Required	**Transmit**	n.a.	n.a.	n.a.
Required	**Remote**	Required	Required	Optional
Required	**Alias_Q**	Required	n.a.	n.a.
Required	**Alias_M**	n.a.	Required	Optional
Required	**Alias_R**	Required	Required	n.a.

Appendix G. System Resources

System set up file: MQFSSET

MQFSSET is a ESDS VSAM file created from a flat file containing system setup information for MQSeries for VSE. MQFSSET is a file used one time only to initialize the System Configuration file, MQFCNFG.

Configuration file: MQFCNFG

MQFCNFG is a KSDS VSAM file which is initialized from MQFSSET at install time by a transaction named MQSU. After installation, it contains only system information such as system constants, system messages, screen messages and the names of CICS maps, programs and transactions. After subsequent updates, any and all user definitions, such as Global System Definition, Queue Definitions and Channel Definitions, are also defined in this configuration file.

MQFCNFG and other resources, including transactions, cannot be used unless a transaction named MQSE (accessing program MQPSENV) is executed. While MQSU is run only once after installation, the transaction MQSE has to be run every time CICS is started. When CICS is started, MQSE must be executed to build a irrecoverable temporary storage area for the purpose of identification.

User definitions are entered and updated by the configuration functions of the Master Terminal (MQMT). The information retained in MQFCNFG, however, is not available for the Queue Manager unless it is brought into storage by executing the MQIT transaction or the initialize option on the operations screen Shutdown of System (function 2.4 of MQMT) or the refresh functions Start/Stop Queue and Open/Close Channel (function 2.1 and 2.2 of MQMT). In other words, after successfully modifying an existing queue definition, the user must stop and refresh this queue (by using the MQMT Start/Stop Queue operation) in order to make these changes available to the Queue Manager.

The other alternative is to execute MQST and then MQIT to shutdown and reinitialize the Queue Manager. However, if new channels or new queues are added, the Queue Manager must be shut down (by MQST or function 2.4 of MQMT) and then initialized (by MQIT or function 2.4 of MQMT) in order to make this new information available to the Queue Manager. Due to the high activity on the MQFCNFG file, it is strongly recommended that it be placed on a DASD volume having a low activity for other files.

Queues

Queues containing application messages, are located in KSDS VSAM clusters. The key to these queues consists of a 48 character object name plus a 4 byte Queue Sequence Number (QSN). Several queues may be defined in the same VSAM cluster. The first record of a queue is a control record. The first 744 bytes of a record contain the message header, not visible from application programs.

The MQSeries Queue Manager uses its own locking facility. It is recommended that the user should not use the CICS ENQ command to obtain exclusive control of queues. Instead, set the MQOO_INPUT_EXCLUSIVE option flag with the MQOPEN command.

Temporary storage

The MQSeries Queue Manager makes wide use of CICS temporary storage. The following temporary storage names are reserved:

- MQSERIES
- MQTAQM
- MQII001
- MQIO001

In-storage-control-blocks and recovery mechanism

It may happen that an application program terminates without issuing the MQDISC command. In such a case, allocated control blocks would never be freed. To overcome this problem, a special task, called System Monitor (MQSM) is started at initialization time.

MQSM has three objectives:

1. Detect and clean unused control blocks.
2. Trigger channel activity when messages have been written into queues.
3. Queue recovery when they become out of sync.

This transaction is then activated at regular time intervals, specified in the global definition “System Wait Interval” (see Figure 14 on page 68).

Appendix H. Sample JCL

Sample JCL to define a configuration file

```
* ** JOB JNM=MQJCONFG,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJCONFG  Define Configuration file for MQ/Series for VSE/ESA.
* ------------------------------------------------------------------*
*      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 ?volid? have also to be modified to suit the   *
*  user specifications.                                              *
*                                                                    *
* ------------------------------------------------------------------*
*                                                                    *
*                                                                    *
*  THIS JOB ALLOCATES A CONFIGURATION FILE FOR MQSERIES FOR VSE/ESA  *
*                                                                    *
*                                                                    *
* ------------------------------------------------------------------*
* Licensed Materials - Property of IBM                               *
*                                                                    *
* 5787-ECX                                                           *
* (C) Copyright IBM Corp. 1993, 1996                                 *
*                                                                    *
* US Government Users Restricted Rights - Use, duplication or        *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp.  *
* ------------------------------------------------------------------*
// EXEC IDCAMS,SIZE=AUTO
 /*                                                                 */
 /*  DELETE VSAM FILES                                              */
 /*                                                                 */
     DELETE (MQSERIES.MQFCNFG) CL NOERASE PURGE  -
       CATALOG(?CAT?)
     SET MAXCC = 0
 /*                                                                 */
 /*  DEFINE VSAM FILE                                               */
 /*                                                                 */
 DEF                                       -
   CLUSTER(NAME(MQSERIES.MQFCNFG)          -
     FILE(MQFCNFG)                         -
     VOL(?volid?)                          -
     RECORDS (300 100)                     -
     RECORDSIZE (2048 2048)                -
     INDEXED                               -
     KEYS(100 0 )                          -
     SHR(2))                               -
   DATA (NAME (MQSERIES.MQFCNFG.DATA) CISZ(4096)) -
   INDEX (NAME (MQSERIES.MQFCNFG.INDEX) CISZ(512)) -
       CATALOG(?CAT?)
 /*                                                                 */
/*
/&
* ** EOJ
```

Sample JCL to define queue file

```
* ** JOB JNM=MQJQUEUE,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJQUEUE Define VSAM clusters for 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 ?volid? have also to be modified to suit the  *
*  user specifications.                                             *
*                                                                   *
* ------------------------------------------------------------------*
*                                                                   *
*                                                                   *
* This job allocates QUEUE files for MQSeries for VSE/ESA.          *
* There is a one-to-one correspondence of the file names in this    *
* job with those in the sample JCL named MQCICSFT                   *
* If there are more files to be allocated, please update MQCICSFT   *
* accordingly.                                                      *
*                                                                   *
* ------------------------------------------------------------------*
* Licensed Materials - Property of IBM                              *
*                                                                   *
* 5787-ECX                                                          *
* (C) Copyright IBM Corp. 1993, 1996                                *
*                                                                   *
* US Government Users Restricted Rights - Use, duplication or       *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp. *
* ------------------------------------------------------------------*
// EXEC IDCAMS,SIZE=AUTO
 /*                                                                 */
 /*  DELETE VSAM FILES                                              */
 /*                                                                 */
    DELETE (MQSERIES.MQFI001) CL NOERASE PURGE  -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFI002) CL NOERASE PURGE  -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFI003) CL NOERASE PURGE -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFO001) CL NOERASE PURGE -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFO002) CL NOERASE PURGE -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFO003) CL NOERASE PURGE -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFLOG) CL NOERASE PURGE -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFERR) CL NOERASE PURGE -
      CATALOG(?CAT?)
    DELETE (MQSERIES.MQFMON) CL NOERASE PURGE -
      CATALOG(?CAT?)
    SET MAXCC = 0
 /*                                                                 */
 /*  DEFINE VSAM FILES                                              */
 /*                                                                 */
```

```
/*                                                                 */
DEF CLUSTER(NAME(MQSERIES.MQFI001)         -
    FILE(MQFI001)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFI001.DATA) CISZ(4096)) -
    INDEX (NAME (MQSERIES.MQFI001.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                                 */
/*                                                                 */
DEF CLUSTER(NAME(MQSERIES.MQFI002)         -
    FILE(MQFI002)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    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?)
/*                                                                 */
DEF CLUSTER(NAME(MQSERIES.MQFI003)         -
    FILE(MQFI003)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFI003.DATA) CISZ(4096)) -
    INDEX (NAME (MQSERIES.MQFI003.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                                  */
/*                                                                  */
DEF CLUSTER(NAME(MQSERIES.MQFO001)         -
    FILE(MQFO001)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFO001.DATA) CISZ(4096)) -
    INDEX (NAME (MQSERIES.MQFO001.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                                 */
DEF CLUSTER(NAME(MQSERIES.MQFO002)         -
    FILE(MQFO002)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFO002.DATA) CISZ(4096)) -
```

```
    INDEX (NAME (MQSERIES.MQFO002.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                              */
DEF CLUSTER(NAME(MQSERIES.MQFO003)         -
    FILE(MQFO003)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 11000)                 -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFO003.DATA) CISZ(12288)) -
    INDEX (NAME (MQSERIES.MQFO003.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                              */
DEF CLUSTER(NAME(MQSERIES.MQFLOG)          -
    FILE(MQFAUDT)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFLOG.DATA) CISZ(4096)) -
    INDEX (NAME (MQSERIES.MQFLOG.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                              */
DEF CLUSTER(NAME(MQSERIES.MQFMON)          -
    FILE(MQFERR1)                          -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFMON.DATA) CISZ(4096)) -
    INDEX (NAME (MQSERIES.MQFMON.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*                                                              */
DEF CLUSTER(NAME(MQSERIES.MQFERR)          -
    FILE(MQFERR)                           -
    VOL(?volid?)                           -
    RECORDS (300 100)                      -
    RECORDSIZE (200 4089)                  -
    INDEXED                                -
    KEYS(52 0)                             -
    SHR(2))                                -
    DATA (NAME (MQSERIES.MQFERR.DATA) CISZ(4096)) -
    INDEX (NAME (MQSERIES.MQFERR.INDEX) CISZ(1024)) -
    CATALOG(?CAT?)
/*
/&
* ** EOJ
```

Sample JCL to define and create the setup file MQJSETUP

```
* ** JOB JNM=MQJSETUP,DISP=D,CLASS=A
* ** LST DISP=H,CLASS=Q,PRI=3
// JOB MQJSETUP - Define/Load Setup file for MQ/Series for VSE/ESA.
* --------------------------------------------------------------------*
*      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"                          *
*                 "*  ** SLI" to  "* $$ SLI"                          *
*                 "*  ** EOJ" to  "* $$ EOJ"                          *
*                                                                     *
*   Fields filed with ?volid? have also to be modified to suit the    *
*  user specifications.                                               *
*                                                                     *
* --------------------------------------------------------------------*
*                                                                     *
*   This job downloads SYSIN.Z from a the sublibray named             *
*   PRD2.MQSERIES to an ESDS VSAM file : MQSERIES.MQFSSET             *
*   This ESDS file is, in turn, the input to MQSU transaction         *
*   to create MQSERIES.MQFCNFG VSAM cluster.                          *
*                                                                     *
* --------------------------------------------------------------------*
* Licensed Materials - Property of IBM                                *
*                                                                     *
* 5787-ECX                                                            *
* (C) Copyright IBM Corp. 1993, 1996                                  *
*                                                                     *
* US Government Users Restricted Rights - Use, duplication or         *
* disclosure restricted by GSA ADP Schedule Contract with IBM Corp.   *
* --------------------------------------------------------------------*
// EXEC IDCAMS,SIZE=AUTO
 /*                                                                  */
 /*  DELETE VSAM FILES                                               */
 /*                                                                  */
     DELETE (MQSERIES.MQFSSET) CL NOERASE PURGE  -
       CATALOG(?CAT?)
     SET MAXCC = 0
 /*                                                                  */
 /*  DEFINE VSAM FILE                                                */
 /*                                                                  */
 DEF CLUSTER(NAME(MQSERIES.MQFSSET) -
     FILE(MQFSSET)                          -
     VOL(?volid?)                    -
     RECORDS (500 100)                      -
     RECORDSIZE (80  80)                -
     NONINDEXED                                -
     SHR(2))                                -
   DATA (NAME (MQSERIES.MQFSSET.DATA) CISZ(4096)) -
     CATALOG(?CAT?)
 /*                                                                  */
/*                                                                  */
// DLBL LOADFL,'MQSERIES.MQFSSET',,VSAM,CAT=MQMCAT
// EXEC IESVSMLD,SIZE=AUTO
80,E,LOADFL
* ** SLI MEM=SYSIN.Z,S=PRD2.MQSERIES
/*
/&
* ** EOJ
```

Glossary

This glossary describes terms used in this book and words used with other than their everyday meaning. In some cases, a definition may not be the only one applicable to a term, but it gives the particular sense in which the word is used in this book.

If you do not find the term you are looking for, see the Index or the IBM Dictionary of Computing, New York: McGraw-Hill, 1994.

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. The ANSI/EIA Standard--440-A: Fiber Optic Terminology.

Copies may be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington DC 20006. Definitions are identified by the symbol (E) after the definition. The Information Technology Vocabulary, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.

A

ADMINISTRATOR COMMANDS. MQSeries commands used to manage MQSeries objects, such as queues, processes and channels.

ALIAS QUEUE OBJECT. An MQSeries object, the name of which is an alias for another queue name. When an application or a queue manager uses an alias queue, the alias name is resolved and the requested operation is performed on the queue with the resolved name.

APAR. Authorized program analysis report.

ATTRIBUTE. One of a set of properties that defines the characteristics of 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.

BROWSE. In message queuing, 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.

C

CHANNEL. See message channel.

CLIENT. The program that requests information in the particular two-program information-flow model of client/server. See also server. In an OS/2, DOS, Microsoft Windows, AIX or UNIX environment, this means a system which supports MQI application programs but does not contain the entire queue manager. For example, several client systems can all logically belong to the same queue manager.

D

DEAD-LETTER QUEUE. A queue to which a queue manager or application sends messages that it cannot deliver to their correct destination.

DISTRIBUTED APPLICATION. In message queuing, a set of application programs that can each be connected to a different queue manager, but that collectively comprise a single application.

DISTRIBUTED QUEUE MANAGEMENT. In message queuing, the setup and control of message channels to queue managers on other systems.

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)

G

GET. In message queuing, to retrieve a message by removing the message from a queue or by browsing the message. See also browse.

I

INPUT PARAMETER. A parameter of an MQI call in which you supply information when you make the call.

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.

L

LOCAL DEFINITION. An MQSeries object that belongs to a local queue manager.

LOCAL DEFINITION OF A REMOTE QUEUE. An MQSeries object that belongs to a local queue manager. This object defines the attributes of a remote queue.

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. To a program, the queue manager to which the program is connected. This is the queue manager that provides message queuing services to that 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.

LOGICAL UNIT OF WORK (LUW). See unit of work.

M

MCA. Message channel agent.

MCAMD. A system program that provides a centralized channel database service allowing MCAs and MQM to access and modify the channel database.

MESSAGE. (1) In message queuing applications, a communication sent from a program to another program. (2) In system programming, information intended for the terminal operator.

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 and a communication link.

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 that is carried as part of an MQSeries message. The format of the message descriptor is defined by the MQMD structure.

MESSAGE QUEUE. Synonym for queue.

MESSAGE QUEUE INTERFACE (MQI). The programming interface provided by the MQSeries message queue managers. This programming interface allows application programs to access message queuing services.

MQSERIES. A family of IBM licensed programs that provides 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. A method for communication between programs. Messaging can be synchronous or independent of time.

MQI. Message Queue Interface.

O

OBJECT. In MQSeries, objects define the attributes of queue managers, queues and process definitions.

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.

P

PERSISTENT MESSAGE. A message that survives a restart of the queue manager.

PLATFORM. In MQSeries, the operating system under which a queue manager is running. See also application environment.

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. Queues can be of type local, alias or remote. Local queues can contain a list of messages waiting to be processed. Queues of other types cannot contain messages -- they point to other 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.

QUEUING. See message queuing.

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.

REMOTE QUEUE. A queue that belongs 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 is remote if it is not the queue manager to which the program is connected.

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 sent.

REQUESTER CHANNEL. In MQSeries, a channel that initiates transfers, communicating with a remote server channel. The requester channel accepts messages from the server channel over a communication link and puts the messages on the local queue designated in the message.

RETURN CODES. The collective name for completion codes and reason codes.

ROLLBACK. Synonym for backout.

S

SENDER CHANNEL. In MQSeries, a channel that initiates transfers, removes messages from a transmission queue, and moves them over a communication link to a receiver channel.

SERVER. 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 MQSeries, 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.

SYNCHRONOUS MESSAGING. A method for communication between programs in which the application waits for a reply before resuming its own processing. Contrast with time-independent 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.

T

TIME-INDEPENDENT MESSAGING. A method for communication between programs in which the requesting program proceeds with its own processing without waiting for a reply to its request. Contrast with synchronous messaging.

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 that allows a queue manager to start an application automatically when predetermined conditions on a queue are satisfied.

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.

U

UNDELIVERED MESSAGE QUEUE. See dead-letter queue.

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 at a user-requested syncpoint. It ends either at a user-requested syncpoint or at the end of a transaction. Compare with unit of recovery.

Index

D

E

F

G

H

I

J

N

O

P

Q

R

S

T

U

V

Sending your comments to IBM

IBM MQSeries for VSE/ESA

User's Guide

SC33-1142-02

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 only 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., use your international access code followed by 44 1962 870229
  - From within the U.K., use 01962 870229
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink: WINVMD(IDRCF)
  - Internet: idrcf@winvmd.vnet.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic number to which your comment applies
- Your name/address/telephone number/fax number/network ID.

Readers' Comments

IBM MQSeries for VSE/ESA

User's Guide

SC33-1142-02

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	
Phone Number	

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.).

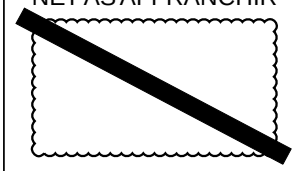
1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCRI NUMBER: PHQ - D/1348/SO

NE PAS AFFRANCHIR



NO STAMP REQUIRED

REPONSE PAYEE
GRANDE-BRETAGNE

IBM United Kingdom Laboratories Limited
Information Development Department (MP 095)
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

Program Number: 5787-ECX

Printed in U.S.A.

SC33-1142-02

IBM MQSeries for VSE/ESA **User's Guide** Version 1 Release 4 SC33-1142-02