MERVA ESA Components

MQI Attachment
User's Guide

Version 4 Release 1

IBM

MERVA ESA Components

MQI Attachment
User's Guide

Version 4 Release 1

First Edition, June 2001

This edition applies toVersion 4 Release 1 of IBM MERVA ESA Components (5648-B30) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About This Book

Read this book to find out how to install, customize, and run MQSeries Interface (MQI) Attachment.

Who Should Read This Book

This book is written for system administrators who want to install and customize MQI Attachment. It is also intended for message-processing personnel who work with messages.

If you are:
- A system administrator, be sure to read Part 1 and Part 2
- Message-processing personnel, be sure to read Part 1 and Part 3

This book assumes that you are familiar with:
- MERVA USE & Branch for Windows NT
- Windows NT
- MQSeries for Windows NT and Windows 2000 Version 5.2

How This Book Is Organized

This book contains the following main parts:
- Part 1 introduces the MQI Attachment architecture. It also provides an overview on how MERVA and a remote application can be linked, and describes MQI Attachment messages, and message processing.
- Part 2 is intended for system administrators. It describes how to install and customize MQI Attachment, and tells you how to customize MERVA USE & Branch for Windows NT and MQSeries. It also discusses the customization of the channel message exit.
- Part 3 describes how to start and work with MQI Attachment, and how to run MQI Attachment as a Windows NT service.

Conventions and Terminology Used in This Book

In this book, the following naming conventions apply:
- MQI Attachment is abbreviated to MQIA. In sections referring to keywords and in the definition samples, MQI Attachment is called MVQAttachment.
- MERVA USE & Branch for Windows NT is abbreviated to MERVA.

Part 1. General Introduction to MQI Attachment (MQIA)

This part provides a general introduction to MQIA. It describes:

- The system architecture of MQIA
- How MERVA and a remote application can be linked
- What an MQIA message is
- How MQIA messages are processed

Chapter 1. Introducing MQI Attachment

MQIA is a part of MERVA. It provides access to MERVA, based on an MQSeries interface. This means MERVA can be accessed within a TCP/IP, or SNA network.

The main tasks of MQIA are:

To exchange messages between a remote application and MERVA
> This lets you use MERVA data on a system on which MERVA is not installed.

To exchange messages between MERVA systems
> For example, this lets you send pregenerated SWIFT session keys from MERVA to MERVA ESA.

System Architecture

Figure 1 shows the architecture of MQIA.
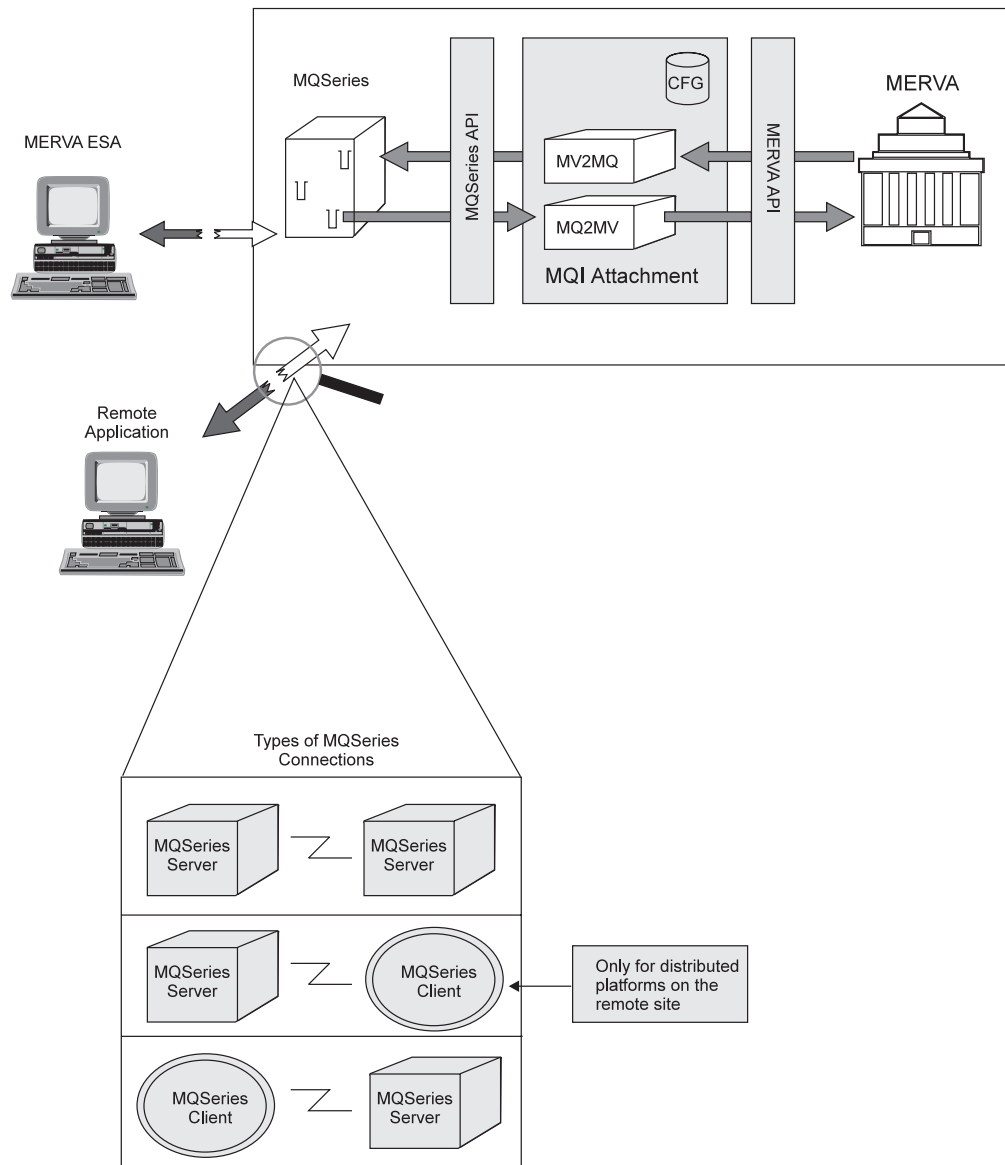
Figure 1. MQIA System Architecture

In Figure 1:
- The MV2MQ process is the MQIA process which processes the message transfer from MERVA to MQSeries
- The MQ2MV process is the MQIA process which processes the message transfer from MQSeries to MERVA

MQIA is a MERVA application that uses the MERVA API to connect to MERVA, and the MQSeries programming interface to connect to MQSeries. MQIA provides:
- Transactional control
- Mapping of MERVA message definitions to MQSeries message definitions, and vice versa

MQIA can be connected to a local MQSeries server, or can use the MQSeries client to connect to a remote MQSeries server.

MQIA can be executed either from the command line, or as a Windows NT service.

Linking MERVA and a Remote Application

A *link* is a connection from MERVA to one remote application. Figure 2 shows the connection between MERVA and an MQSeries queue manager for one link.
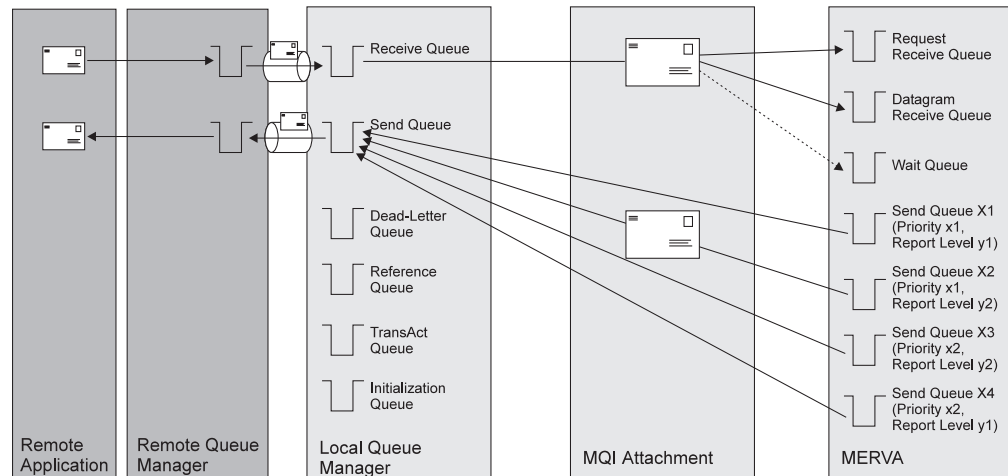
Figure 2. The Connection between MERVA and an MQSeries Queue Manager

MQIA connects MERVA to a MQSeries queue manager. The basic requirements for such a connection are one send and one receive queue on each side. This is, however, not sufficient, because the MERVA and MQSeries message definitions are different. Within MQSeries, more options are specified with a message than within MERVA; for example, report options, message type, priority. For this reason, several send queues in MERVA are mapped to one send queue in MQSeries, and two receive queues in MERVA are mapped to one receive queue in MQSeries. Each MERVA send queue is associated with a priority level, report level, and field option. When a message arrives in a MERVA send queue, the message is transferred to the MQSeries send queue, and the associated message options are set in the message's message descriptor.

It is not possible to identify message types within MERVA; for example, whether a message is a request or datagram. For this reason there are two receive queues in MERVA: One for incoming request messages, and one for datagrams.

As shown in Figure 3 on page 6, you can connect more than one remote application via MQIA to MERVA, because MQIA supports more than one link. Each link has its own send and receive queues in both MQSeries and MERVA. The following queues are shared between all links on the MQSeries side:
- Reference queue
- Dead-letter queue
- TransAct queue
- Initiation queue (trigger)

On the MERVA side, each link has its own MERVA wait queue.

Figure 3 shows a multi-link connection.
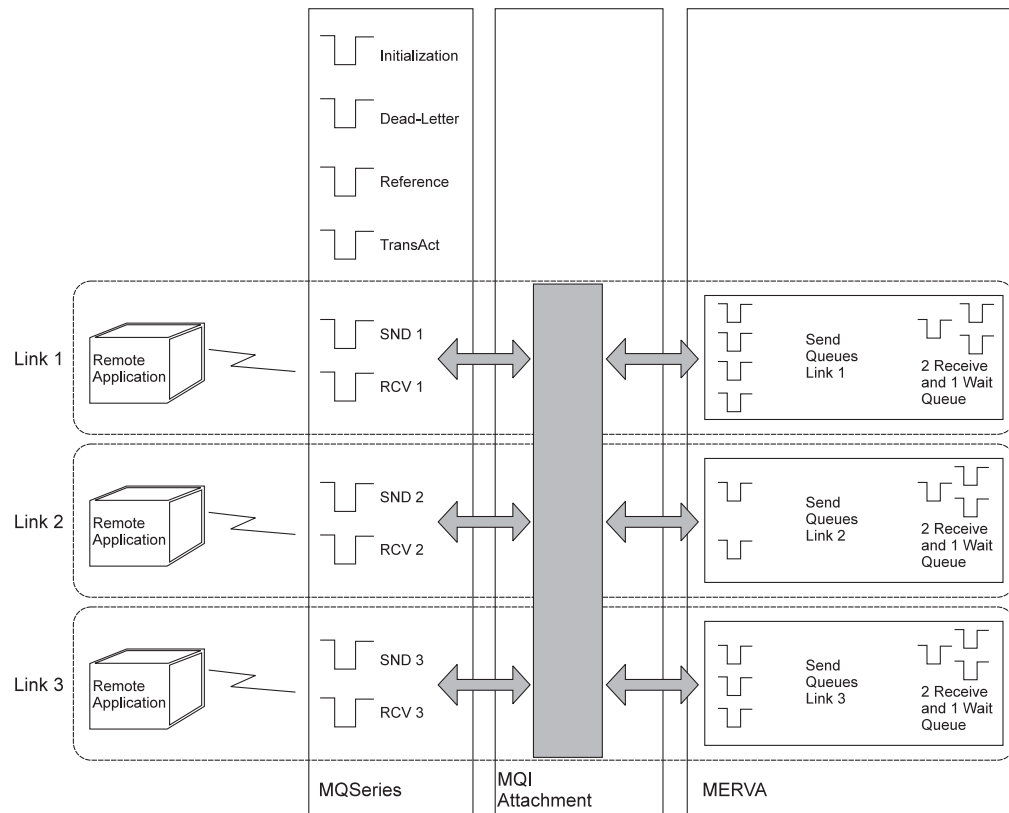
Figure 3. MQIA Multi-Link Connection

MQIA monitors the arrival of a message in any MQSeries receive queue via an
MQSeries initiation queue (trigger queue). MQIA monitors the arrival of a message
in the MERVA send queue via the MERVA semaphore technique. See the *MERVA
Application Programming Guide*.

Purpose Description of the Queues Used by MQIA

MQSeries Send Queue
This queue is used to send messages to the remote application via
MQSeries.

MQSeries Receive Queue
This queue is used to receive all incoming messages, and reports for an
MQIA link. The queue must be enabled for triggering on every arriving
message.

MQSeries Reference Queue
This queue is used to share reference messages for correlation:
- Requests with replies and reports
- Datagrams with reports

MQSeries Dead-Letter Queue
All messages that generate an error are placed in this queue. For example,
messages that MERVA rejects, or reply messages to a non-existent request.

MQSeries TransAct Queue
This queue is used by MQIA transaction processing. This processing

ensures that no message is lost, and no message is duplicated in error situations. If no special TransAct queue is specified, the MQSeries reference queue is used as the TransAct queue.

MQSeries Initiation Queue (Trigger Queue)
The queue manager creates a trigger message for every incoming message in every receive queue that is assigned to the initiation queue. The initiation queue is also known as trigger queue in MQIA. MQIA uses this queue to monitor all MQSeries receive queues for the arrival of a message. When a trigger message arrives, MQIA gets it, and retrieves the initiating receive queue name from the message. It then gets the message from the queue with the retrieved queue name.

MERVA Send Queue
MQIA transfers messages arriving in this queue to the MQSeries send queue. The message characteristics (for example, message type and report level) are defined by the send queue configuration, not within the message.

MERVA Request Receive Queue
All incoming request messages from the remote application are put in this queue. This enables MERVA to differentiate between requests and datagrams. The receive queue type is defined in the configuration file.

MERVA Datagram Receive Queue
Receives datagram messages from the remote application.

MERVA Wait Queue
All messages that MERVA sends to the remote application, and that are waiting for a response (reply, reports, or both), are routed to the wait queue. A message is routed out from the wait queue only if all required responses have been received. The dotted line between the MQSeries receive queue and the MERVA wait queue in Figure 2 on page 5 shows that there is also a connection between the MQSeries receive queue and the MERVA wait queue. The MERVA wait queue is not a receive queue in the same way as the request receive and datagram receive queues. This is because it only updates the waiting messages with received report and reply messages.

The current state of a message in a MERVA wait queue can be identified. To do this, the MERVA field **MSGACK** is used as a temporary buffer for the required outstanding reports and replies. The original value is also stored in the corresponding message on the MQSeries reference queue. It is restored when the required report level is reached.

MQIA Error Handling

When an error occurs, MQIA creates the appropriate error message by attaching the original message to the dead-letter queue header. The reason field in this dead-letter queue header contains the reason code that explains why the message was put in the dead-letter queue, see Table 4 on page 61.

Chapter 2. MQIA Messages

This chapter describes:
- The type of MQSeries messages that MQIA recognizes
- The MQIA message layout

Message Types

MQIA recognizes the following message types:

Datagram Message
> A datagram message is a message that does not require a reply from the application. MQIA sends the following types of datagrams:

> **Datagram with Report Options**
>> These are MQSeries specific message options that request a status report, from the remote queue manager, about the sent message, see "Report Message". MQIA processes the following report options only:
>> - Confirm on Arrival (COA)
>> - Confirm on Delivery (COD)
>> - Exception report
>>
>> All other reports are treated as invalid, and put in the dead-letter queue.

> **Datagram without Report Option**
>> No report or reply is expected (send and forget).

Request Message
> The sender of a request message expects a reply from the remote application, and waits until this reply message arrives. A request message can also require additional reports as described in "Datagram Processing Initiated by a Remote Application" on page 19.

Reply Message
> A reply message should only be sent in response to a request. If it is not, it cannot be assigned to a request, and MQIA discards it, and puts it in the dead-letter queue. A reply from a remote application should only contain those fields that are defined in the MERVA API. If other fields are included, the message has the wrong message format, and MQIA puts it in the dead-letter queue.

Report Message
> A report message reports the status of a sent message, and is created by the MQSeries queue manager, or an application. Report messages accepted by MQIA indicate the following:
> - Whether a sent message has arrived in the remote queue manager. This is confirm on arrival (COA).
> - Whether a sent message has been delivered to the remote application. This is confirm on delivery (COD).
> - Whether an error (exception) has occurred in the MQSeries queue manager.
> - Whether an error (exception) has occurred in the remote application.

Information on outstanding reports is also included in the MERVA field **MSGACK** in the corresponding message in the MERVA wait queue. This makes it possible for you to verify in which state the message currently is.

Message Layout

An MQIA message consists of a sequence of base elements. There are several layout types, and they are derived depending on the sequence (see "Layout Types" on page 11).

Base Elements

An MQIA message comprises the following base elements:

Application Message

Valid application messages are MERVA messages, like the following:
- SWIFT FIN messages
- Telex messages
- User-defined messages

Length Field (four bytes)
This field contains the length, in bytes, to the next length field, or to the message end. It includes the four bytes of the length field itself.

Field Identifier (eight characters)
The field identifiers must be the same as the MERVA-related data identifiers of MERVA. Identifiers shorter than eight characters are filled with blanks to make them eight characters long. MQIA treats messages with field identifiers other than MERVA-related data field names as invalid, and puts them in the MQSeries dead-letter queue. The exception is the eight blanks identifier, which is a valid identifier for the application message of a reply message.

MQIA allows you to define which MERVA-related data fields are to be sent with the message. The sending of an application message can only be disabled for replies. For datagram and requests, the application message is always sent with the message. If there are multiple occurrences of the same data field in a message, MQIA uses the last occurrence only. All others are ignored.

Field Value (variable length)
This field is also referred to as the data field. It contains either the values of a MERVA-related data field or the application message. The field value is always a string, not a number. The length is variable, and in almost all cases specified by the length field preceding the field identifier. The maximum length of a value string depends on the field type, (see *MERVA Application Programming Interface Guide*). If a string value exceeds the maximum length, MQIA truncates it as required.

MERVA-Related Fields

The following MERVA-related fields are valid for MQIA messages:
- **MSGOK**
- **MSGACK**
- **MSGROUTE**
- **MSGCMNT**
- **MSGMAC**

- **MSGPAC**
- **TXHEAD**
- **TXINFO**

The other MERVA-related fields that are available within MERVA are not valid for MQIA messages. You can select which fields must be sent with a message with the MERVA send queue definition. The exception is **MSGNET**, which is always sent with the message.

Table 1 shows the relationship between the MERVA-related field names and the corresponding field ID used within a MQIA message. The maximum value length shows the maximum length of the field data. Field IDs that are shorter than eight characters are filled with blanks to make the ID eight characters long. In Table 1 the blanks are shown as '_'.

The field **MSG** contains the application message. It is used with field ID in reply messages only.

The field **MSGNET** specifies the network type of the message. It is always mapped to the first character of the field **ApplIdentityData** in the message descriptor of the MQSeries message as follows:
- A SWIFT-FIN message to a 'W' character
- A telex message to a 'P' character
- A user-defined message to a 'U' character

Table 1. MERVA-Related Field Names

MERVA-related Field	Field ID in MQIA Message	Maximum Value Length
MSGOK	″**MSGOK___**″	8
MSGACK	″**MSGACK__**″	127
MSGROUTE	″**MSGROUTE**″	19
MSGCMNT	″**MSGCMNT_**″	1,999
MSGMAC	″**MSGMAC__**″	127
MSGPAC	″**MSGPAC__**″	127
TXHEAD	″**TXHEAD__**″	587
TXINFO	″**TXINFO__**″	145
MSG	″_____″	28,000
MSGNET	**ApplIdentityData[0]**	1

Layout Types

MQIA recognizes the following layout types:

Layout Type 1
An MQIA message that consists of the application message. This message layout is only valid for datagram and request messages.

Application
Message

Figure 4. MQIA Message Consisting of an Application Message

Layout Type 2

An MQIA message that consists of only one data field, preceded by a field identifier. The data field can be a MERVA-related field, or an application message. This message layout is only valid for reply messages.

Field ID	Field Value

Figure 5. MQIA Message Consisting of One Data Field Preceded by a Field Identifier

Layout Type 3

An MQIA message consisting of an application message and at least one MERVA-related data field. The application message is preceded by a length field, and has no field identifier. It must be the first data field in the message. This message layout is only valid for datagram or request messages.

Length App. Msg	Application Message	Length Field 2	Field-ID Field 2	Field Value Field 2	...	Length Field n	Field-ID Field n	Field Value Field n

Figure 6. MQIA Message Consisting of One Application Message and at least One MERVA-related Data Field

Layout Type 4

An MQIA message consisting of more than one data field which can be with or without an application message. The application message can be anywhere within the MQIA message, but it is usually the first data field in the message. Any data field is preceded by a length field and a field identifier. MQIA uses eight blanks as the field identifier for the application message. All the other identifiers are the same as those described in the *MERVA Application Programming Interface Guide*. This message layout is only valid for reply messages. Figure 7 shows a sample layout that includes an application message.

Length Field 1	_____	Application Message	Length Field 2	Field-ID Field 2	Field Value Field 2	...	Length Field n	Field-ID Field n	Field Value Field n

Figure 7. MQIA Message Consisting of More Than One Data Field and One Application Message

Layout Type 5

An MQIA message with length zero. Valid only for reply messages.

Table 2 shows the relationship between MQIA message types, and MQIA message layouts.

Table 2. Valid MQIA Layouts and their Relationship to MQIA Message Types

Message Type	String Message	Non-String Message
Datagram or Request	Layout type 1: contains only the application message.	Layout type 3: contains the application message, and at least one MERVA-related data field.

Table 2. Valid MQIA Layouts and their Relationship to MQIA Message Types (continued)

Reply	Layout type 2: contains only one data field. This can be the application message, or any MERVA-related data field.	Layout type 4: contains more than one data field which can be the application message, and any one MERVA-related data field, or more than one MERVA-related data field.
	Layout type 5: reply message with length zero.	

Note: MQSeries recognizes a message as a string message if the message descriptor format field contains any identifier starting with ″M2″. This means:

- Layout types 1, 2, and 5 are string messages.
- Layout types 3 and 4 are non-string messages.

Chapter 3. How MQIA Processes Messages

This section describes how MQIA processes:

- A request message sent by MERVA
- A request message sent by a remote application
- A datagram message sent by MERVA
- A datagram message sent by a remote application

Request Processing Initiated by MERVA

Before you can send a request message from MERVA to a remote application, you must define the appropriate MERVA send queue as a request queue for MQIA. You define this in the MQIA configuration file.

In Figure 8, Figure 9 on page 16, and Figure 10 on page 17, the numbers refer to the steps described in the section that follows each figure.

Figure 8. Request Processing Initiated by MERVA Part 1

1. MERVA routes a message to the appropriate MERVA send queue, or another application puts a message directly in the MERVA send queue.
2. MQIA gets the message from the MERVA send queue.
3. Processing:
   a. MQIA checks if a message with the same message ID as the message reference number (MRN) of the request message is already in the reference queue. If so, and if the message is from an incoming request, MQIA does the following:
      - Automatically changes the message type from request to reply.
      - Fills the reply message with only those fields defined in the configuration file. See "Request Processing Initiated by a Remote Application" on page 17.

      If the message in the reference queue does not come from an incoming request, it is put in the dead-letter queue, with the appropriate reason code, (see Table 4 on page 61). If the MRN is not found in the reference queue, a new reference message is created in the reference queue. The MRN of the

request message becomes the message ID. This new message contains the original **MSGOK**, the **MSGACK** field contents, and the required report level.

b. After the reference message has been created, MQIA does the following:

- Sets the **MSGOK** field of the original request message in the MERVA send queue to ″**MVQWAIT**″
- Sets the **MSGACK** field to the required report options
- Routes the request message according to its routing conditions

c. The routing conditions for all MERVA send queues must be defined in such a way that, if the value in the **MSGOK** field is ″**MVQWAIT**″, the message is routed to the MERVA wait queue. On any other value the message is routed to any target queue other than the original MERVA send queue. This avoids routing loops.

4. MQIA puts the message in the MQSeries send queue.

5. The queue manager transfers the message to the remote queue manager. From there it is routed to the remote application. Alternatively the remote application gets the message from the MQSeries send queue via an MQSeries client connection.
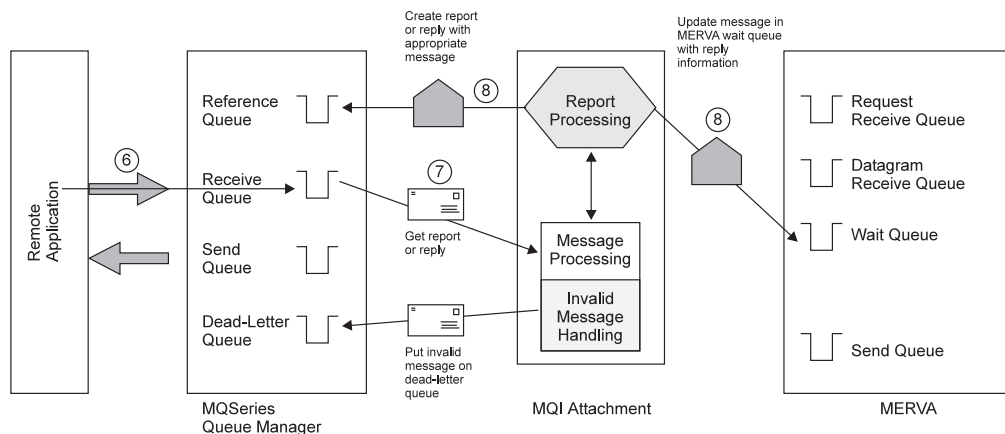
Figure 9. Request Processing Initiated by MERVA Part 2

6. The queue manager receives a message from the remote application. This message is put in the MQSeries receive queue.

7. MQIA receives a trigger message created by the queue manager, and gets the original message from the appropriate receive queue.

8. MQIA determines the message type of the received message. If it is a report or a reply message, MQIA checks whether the corresponding request or datagram message is in the MQSeries reference queue.

- If it is a report message, MQIA updates the report level that the message has reached in the reference queue, and in the MERVA wait queue. It removes the report message from the MQSeries receive queue.

- If it is a reply message, MQIA inserts the message contents, and the contents of any MERVA-related fields contained in the reply message, into the original request message. The original request message is waiting in the MERVA wait queue.

- If the corresponding request or datagram message is not found in the MQSeries reference queue, MQIA error handling is performed, see "MQIA Error Handling" on page 7.

- If it is a datagram, see "Datagram Processing Initiated by a Remote Application" on page 19.
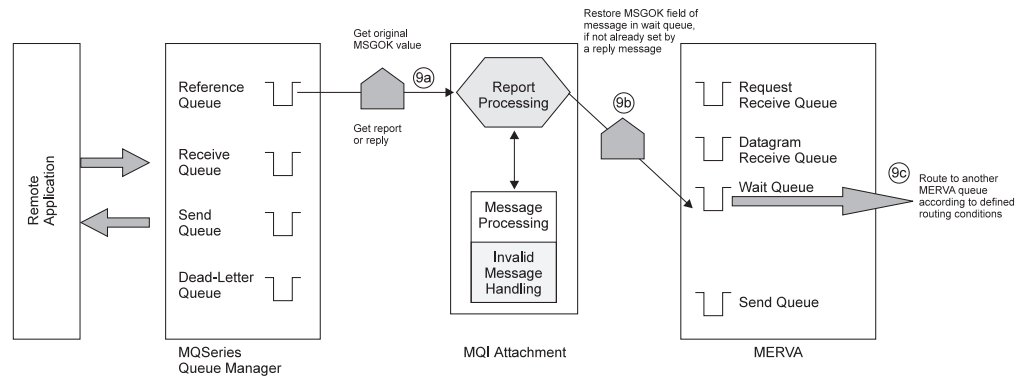
Figure 10. Request Processing Initiated by MERVA Part 3

9. If no **MSGOK** field has been sent within the reply message:
   a. MQIA gets the original **MSGOK** and **MSGACK** field contents from the message in the MQSeries reference queue.
   b. MQIA restores the **MSGOK** and **MSGACK** field contents of the original request message in the MERVA wait queue, with the previously retrieved original value depending on the reply message fields.
   c. The message is removed from the reference queue, and the message in the MERVA wait queue is then routed according to the routing conditions defined for the wait queue. The request has now been processed.

Request Processing Initiated by a Remote Application

This section describes how an incoming request message from a remote application is processed.

In Figure 11, and Figure 12 on page 18, the numbers refer to the steps described in the section that follows each figure.
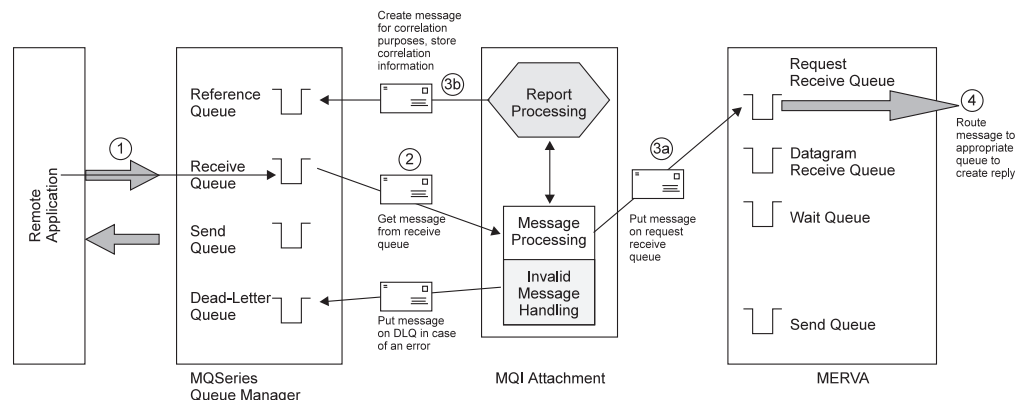
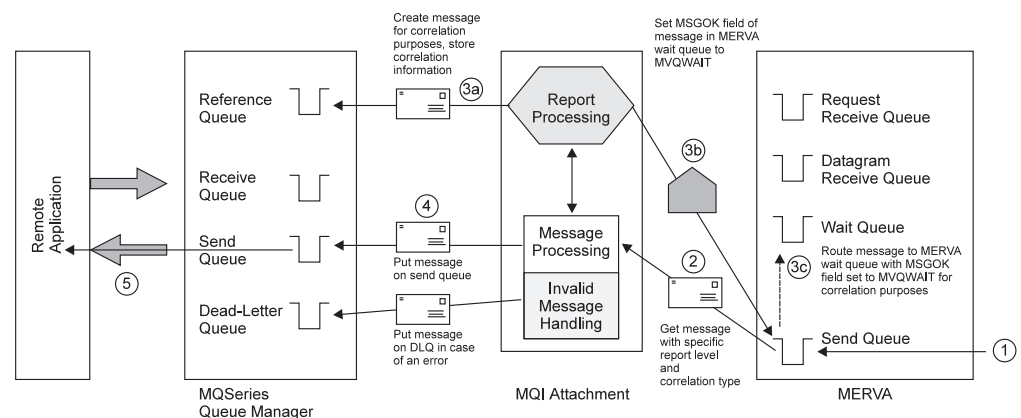*Figure 11. Request Processing Initiated by a Remote Application Part 1*

1. A message arrives from either the remote application via a remote queue manager, or via a MQSeries client connection in the local queue manager. It is put in the MQSeries receive queue.

2. MQIA receives a trigger message that a new message has arrived, and gets the message from the MQSeries receive queue. It determines the message type.
3. Processing:
   - Report and reply message processing is described in "Request Processing Initiated by MERVA" on page 15
   - Datagram processing is described in "Datagram Processing Initiated by a Remote Application" on page 19
   a. In the case of a request message, MQIA creates a new message in the MERVA request receive queue.
   b. After MQIA has created the message in the MERVA request receive queue, it creates a new message in the MQSeries reference queue. This means that the message ID of the message in the reference queue must be the same as the MRN of the message previously created in the MERVA request receive queue. The queue manager automatically handles any additional reports (for example, COA or COD) that the remote application requests.
4. The original request message must be used to create a reply message. This is because the reply must have the same MRN as the request message in the MQSeries reference queue. If it does not, it is handled as a request or datagram instead of as a reply message, depending on the MERVA send queue definition in the configuration file. The request message must be routed to another MERVA queue where it is processed by a user or another process, or it must be routed directly, as the reply message to a MERVA send queue of the same link. You must set the routing conditions for the request receive queue to route the messages to the appropriate queue(s); for example, the **SMEDIT** queue.
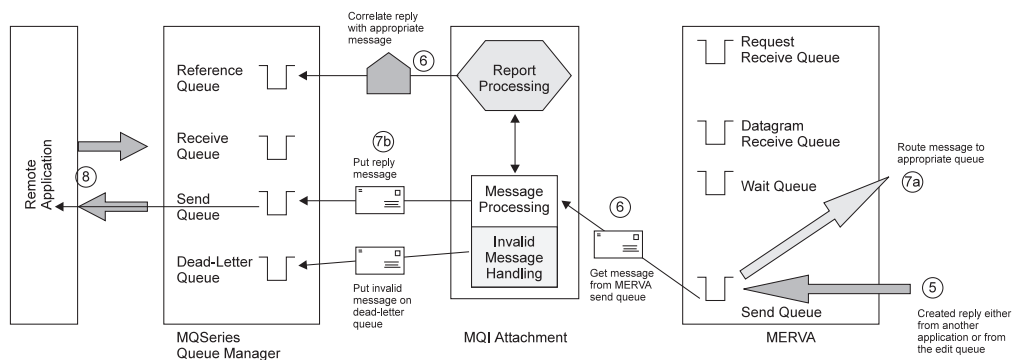
Figure 12. Request Processing Initiated by a Remote Application Part 2

5. A message arrives in the MERVA send queue.
6. MQIA gets the message from the MERVA send queue, and checks whether it is a reply message. It checks this by searching the MQSeries reference queue for a message with the same message ID as the MRN of the arrived message. If MQIA finds a message, and the message is from a request message, MQIA treats the new message as a reply to that request. Otherwise the message is handled as a request or datagram, depending on the MERVA send queue configuration.
7. Processing:
   a. The original message in the MERVA send queue is routed to the target queue according to the routing conditions defined in MERVA.
   b. MQIA puts the reply message in the MQSeries send queue, and removes the corresponding request message from the reference queue.

8. The MQSeries queue manager sends the message to the remote application via the configured paths. The request is processed.

Datagram Processing Initiated by MERVA

This section describes how an incoming datagram message from MERVA is processed.

In Figure 13 the numbers refer to the steps described in the section that follows the figure.
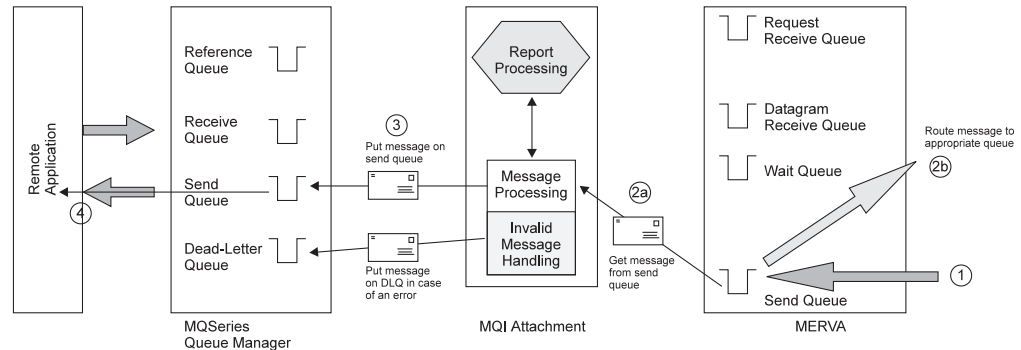
Figure 13. Datagram Processing Initiated by MERVA

1. A message arrives in the MERVA send queue.
2.
    a. MQIA gets the message from the MERVA send queue. It checks in the MQSeries reference queue to see if a message with the same message ID as the new message's MRN already exists. If no such message exists, and if the MERVA send queue is not configured as a request queue, the message is treated as datagram. If the MERVA send queue is configured with some report options, MQIA handles the message as a request message. The only difference is that no reply from the remote application is expected. The processing is the same as described in "Request Processing Initiated by MERVA" on page 15.
    b. If no report options are required, MQIA routes the arrived message, according to the routing conditions defined in MERVA.
3. MQIA puts the message in the MQSeries send queue.
4. The MQSeries queue manager sends the message to the remote application via the configured paths. The datagram processing is completed.

Datagram Processing Initiated by a Remote Application

This section describes how an incoming datagram message from a remote application is processed.

In Figure 14 on page 20, the numbers refer to the steps described in the section that follows the figure.
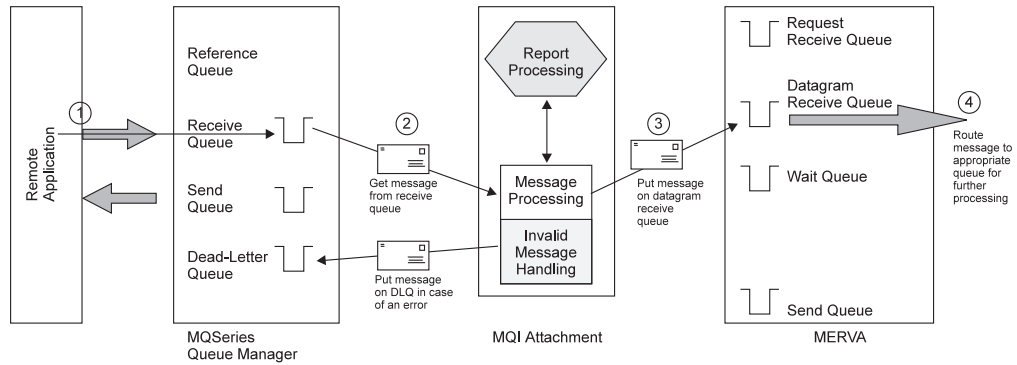
Figure 14. Datagram Processing Initiated by a Remote Application

1. A message arrives in the local queue manager, and is put in the MQSeries receive queue.
2. MQIA gets the message from the receive queue, and determines the message type.
3. If the message is a datagram, MQIA puts the message in the MERVA datagram receive queue and initiates a MERVA routing.
4. The message is routed to the appropriate queue according to the routing conditions defined in MERVA. The datagram processing is completed.

Message Correlation Modes

MQIA supports the following MQSeries correlation modes for incoming requests:
- MQRO_NEW_MSG_ID - MQRO_COPY_MSG_ID_TO_CORREL_ID
- MQRO_PASS_MSG_ID - MQRO_PASS_CORREL_ID

MQIA supports the following MQSeries correlation mode for outgoing datagram, and request messages:
- MQRO_NEW_MSG_ID - MQRO_COPY_MSG_ID_TO_CORREL_ID

For further information, see *MQSeries Application Programming Guide*.

Part 2. Installation and Customization

This part describes the software and hardware required to install MQIA. It also shows the procedures that are necessary to customize MQIA, your operating system, and MQSeries.

Chapter 4. Hardware and Software Requirements

This chapter lists the hardware and software requirements that are necessary to install MQIA.

Hardware Requirements

The hardware requirements are described in the following documents:
- *MQSeries Planning Guide*
- *MERVA USE & Branch for Windows NT Installation and Customization Guide*

Software Requirements

The software requirements for MQIA are:
- Windows NT 4.0, Service Pack Level 5 or higher
- MERVA ESA Components:
  - MERVA USE & Branch for Windows NT, component ID 5648–B30, PTF level UQ52508
- MQSeries for Windows NT Version 5.1 or higher, component ID 04L1830, CSD level CSD04 or higher

Installed Files

MQIA is automatically installed when you install MERVA. During installation, the following files are placed in subdirectories of the MERVA base directory (for example, **c:\merva\use_branch**):

Fully-Qualified File Name
> **Description**

***MERVA base directory*\BIN\CMEMQAT.EXE**
> The MQIA executable file.

***MERVA base directory*\DOC\CMEMQAT.INF**
> An online version of the *MQI Attachment Messages and Codes*.

***MERVA base directory*\SAMPLES\MQI_Attachment\MVQTRXIT.C**
> The sample code for the MQSeries message channel exit.

***MERVA base directory*\SAMPLES\MQI_Attachment\MVQTRXIT.MAK**
> Is the make file to compile and link the MQSeries message channel exit.

***MERVA base directory*\SAMPLES\MQI_Attachment\MVQTRXIT.DLL**
> Is the sample MQSeries message channel exit.

***MERVA base directory*\SAMPLES\MQI_Attachment\MQCONF.MQ**
> Is a sample file that defines all MQSeries objects needed by MQIA. You can change the files according to your needs.

***MERVA base directory*\SAMPLES\MQI_Attachment\CMEMQAT.ATN**
> Is a sample configuration file for MQIA. You can modify the file to meet your needs, then use it as the default configuration file for the corresponding MERVA instance. To do this, copy it to the MERVA instance directory ***MERVA base directory*\INSTANCES**. Then rename it to

instance.**ATN**, where *instance* is the name of the corresponding MERVA instance. You can also copy the file to a directory of your choice to use it as an individual configuration file.

MERVA base directory**BIN****CMENYMIG.REX**
Migration script that migrates profiles from the former MQIA version 4.1.0 to the current format.

MERVA base directory**BIN****CMENEMQE.DLL**
Encryption and decryption DLL for using MQIA channel message exit, or as MQIA send and receive exit. This DLL performs the encryption and decryption of a message on the wire. It is compatible with the version provided with MERVA ESA Version 4.2.

Note: If you want to use this exit, make sure that the group "MQM", or "everyone", is in the DLL access list.

MERVA base directory**BIN****CMENYPEC.EXE**
Password encryption utility.

Chapter 5. Customizing MERVA and MQSeries for MQIA

This chapter describes how to customize MERVA and MQSeries for MQIA.

Customizing MERVA

To customize your MERVA system for MQIA, you must:

Message Queues
- Create MERVA send queues within the MERVA API purpose group
- Create MERVA receive queues within the MERVA API purpose group
- Create MERVA wait queue within the MERVA API purpose group

Alarms and Semaphores
Assign an alarm to each MERVA send queue.

Message Routing

Define the routing conditions for the MERVA send queues:
- Route to the MERVA wait queue if **MSGOK = ″MVQWAIT″**
- Route to the final target queue if **MSGOK <> ″MVQWAIT″**

Define the routing conditions for the MERVA receive queues:
- Route to the final target queue if **MSGOK <> ″MVQWAIT″**
- Or leave it in the receive queue

Define the routing conditions for the MERVA wait queue:
- Route to the final target queue if **MSGOK <> ″MVQWAIT″**
- Or leave it in the wait queue

User ID and Password
Define the MERVA user ID and password.

Access Rights
Define the MERVA user access rights to the previously defined MERVA queues for MQIA.

Note: Ensure that all ready messages are routed to a target queue other than the one used by MQIA. This is because leaving messages on the MQIA queue extends the MQIA startup time, as MQIA performs its recovery processing during startup. This recovery processing checks all MQIA queues for possible ready messages to route to their destinations.

For detailed information on how to do this, refer to the *MERVA USE & Branch for Windows NT Installation and Customization Guide*.

Customizing MQSeries

The following MQSeries objects must be created or defined, and customized:
- Create MQSeries queue manager
- Create MQSeries send queues
- Create MQSeries receive queues
- Create MQSeries initiation queue (trigger queue)
- Create MQSeries reference queue

- Create MQSeries transaction queue
- Create MQSeries process definition
- Define MQSeries send channels
- Define MQSeries receive channels
- Define MQSeries client and server connection channels if using the MQSeries client

To create the queue manager that you want to use, do the following:

1. Start MQSeries explorer. Select:

   Start -> Programs -> IBM MQSeries -> MQSeries Explorer.
2. Select the queue manager item in the tree, and click the right mouse button.
3. Select **New -> Queue Manager**. The Queue Manager creating window (Step 1) is displayed.
4. Enter the new queue manager name; for example **QMGR1**.
5. Enter the default transmission, and dead-letter queue name; for example **DXQ** and **DLQ**.
6. Click the **Next** button. The Queue Manager creating window (Step 2) is displayed. Adapt the protocol path to your needs.
7. Click the **Next** button. In the Queue Manager creating window (Step 3), select **Channel for server connection...**.
8. Click the **Next** button. The Queue Manager creating window (Step 4) is displayed.
9. If TCP/IP is the preferred network protocol, specify the port number for the TCP/IP listener, for example **1414**.
10. Click **Finish** to create the queue manager.

After you have created and started the queue manager, you must create the MQSeries objects that are used by MQIA. If you are using the configuration script that is provided with MQIA, do the following:

1. Switch to a command prompt window.
2. Create a working directory in a drive of your choice, for example, **D:\MQIA**.
3. Change to the working directory.
4. Copy the configuration script file **mqconf.mq** from the directory
   MERVA base directory\Samples\MQI_Attachment into the working directory.
5. Modify the configuration script file as required.

   Note: Set the queue names of the default transmission queue, and dead-letter queue to the names you specified when you created the queue manager.
6. Enter the following command to create the MQSeries objects that MQIA needs to run the modified script file:

   ```
   runmqsc QMGR1 < mqconf.mq > myconf.out
   ```

   Where:

   QMGR1　　　　The previously created queue manager.

   mqconf.mq　　The previously copied and modified configuration script file.

   myconf.out　　The text file created by MQSeries that contains the results of the configuration script execution.
7. Check the results text file for errors.

Configuring the Channels to Start Automatically

To configure the channels to start automatically when new messages arrive in the transmission queue, do the following:

1. Open the appropriate queue manager queue's folder in MQSeries explorer.
2. Select the transmission queue to the remote system; for example, **DXQ**.
3. Click the right mouse button to open the context menu.
4. Select the **Properties** menu item.
5. Select the triggering notepad.
6. Set the trigger control to **ON**.
7. Enter the send channel name in the trigger data field; for example, **LOCAL.TO.REMOTE**.
8. Select **SYSTEM.CHANNEL.INITQ** as the initiation queue.
9. Click **OK** to save the settings.

Connectivity

You can use the following network protocols:

- TCP/IP
- SNA LU 6.2
- NetBIOS
- SPX

The following section describes how to configure TCP/IP, as this is the most commonly used network protocol. For a description of how to configure the other protocols, see the *MQSeries Planning Guide*.

Customizing TCP/IP for MQSeries

MQSeries provides its own TCP/IP listener. To run the listener automatically when the queue manager starts, do the following:

1. Open the MQSeries Services explorer.
2. Select the appropriate queue manager. All the available services for that queue manager are displayed in the right window.
3. If no listener is displayed, open the context menu, and select **New listener**.
4. Select the **Parameter** notepad, and enter a valid port number; for example, **1414**.
5. Select the **General** notepad, and switch the start type to **Automatic**.
6. Click **OK** to save the settings.

To start the send channels automatically, see "Configuring the Channels to Start Automatically".

Chapter 6. Customizing MQIA

To make a connection between MERVA and MQSeries, you must provide an MQIA configuration file. How to define such a configuration file is described in the section "The Configuration File". How to migrate a previous version of the configuration file to a current configuration file is explained in "How to Migrate from a Previous MQIA Configuration File" on page 35.

To invoke MQIA from the Windows desktop, you must create an MQIA desktop program object, see "Creating the MQIA Desktop Program Object" on page 35.

To invoke MQIA as a Window NT service, you must install an MQIA service instance. See "Installing MQIA Service" on page 46 for information on how to install and delete an MQIA service instance.

To make the connection between MERVA and MQSeries, you must define the sections in the configuration file as described in this chapter.

The Configuration File

The configuration file influences the behavior of MQIA, with respect to MERVA and MQSeries. It contains the names and parameters required to run MQIA.

The configuration consists of the following sections:

Common Section
General definitions that apply to all MQIA links. These definitions should exist only once.

Link-Specific Section
Definitions that apply to a specific MQIA link. These definitions must be repeated for each MQIA link.

MERVA provides a sample configuration file, **cmemqat.atn**. You can find this file in the directory **MERVA base directory\samples\MQI_Attachment**.

A copy of the file is included in "Appendix B. Sample Configuration File CMEMQAT.ATN" on page 59.

Syntax Rules

The following syntax rules apply for the configuration file definitions, and conform to the Windows .INI file syntax:

- A stanza consists at least of a keyword and a delimiter. The delimiter is the equal sign (=). If the keyword requires a value, the value should succeed the delimiter.
- You can write keyword, delimiter, and value without blanks, or separate them by any number of blanks.
- You can write keywords and values in lower- or uppercase.
- Values are used in the same form as they are stored.
- You can use blank lines and comment lines.
- A comment starts with a number sign ″#″, or ″;″. Text that follows is ignored.

- A comment can succeed a stanza.
- You cannot split a stanza over two lines.
- The maximum length for MQSeries resources, such as the name of the queue manager, and the queues, is 48 characters.
- The maximum length for directory names is 128 characters.
- The maximum length for other resources, such as attachments, links, MERVA queues, and semaphores, is 8 characters.

Because the configuration file layout is the same as the Windows .INI files, you can also define all configuration data in the Windows NT registry database.

Common Section Parameters

The section name is **[MVQAttachment]**.

You can specify the following parameters:

APITrace
> Optional parameter. Switches the MERVA API trace option on or off. Valid values are **YES** and **NO**. The default is **NO**. If **YES**, MERVA writes additional trace information for the MERVA API calls that MQIA uses.

LogDir
> MQIA logging directory. Mandatory parameter. Specifies the directory to which the MQIA log file is written.

LogLevel
> MQIA log level. Optional parameter. Valid values are from 1 to 4. The default is 1. The type of information, and log level recorded are:
> - Errors only (log level 1)
> - Errors, and warnings (log level 2)
> - Errors, warnings, and successful operation information (log level 3)
> - Debug information (log level 4)

MQClient
> Optional parameter. Specifies whether MQIA should connect to the MQSeries queue manager via an MQSeries client connection. Valid values are **YES** and **NO**. The default is **NO**.

MQQMgrName
> MQSeries queue manager name. Mandatory parameter. Defines the name of the MQSeries queue manager. This can be a local queue manager or a remote queue manager, depending on the key MQClient. If a client connection table is used, the queue manager name can also be a queue manager group, identified by the wildcard character '*' at the beginning of the queue manager name. MQIA then retrieves the real queue manager name of the connected queue manager, and displays it in the log file (if the log level >= 3).

MQPIIRecovery
> Enable/disable polling mode of the MQ2MV process for all links. Optional parameter. The default is **NO**. Valid values are **YES** and **NO**.

MQRefQName
> MQSeries reference queue name. Mandatory parameter. Defines the global reference queue name.

MQTActQName

MQSeries TransAct queue name. Optional parameter. If this queue is not specified in the configuration file, the MQSeries reference queue is also used as the TransAct queue.

MQTrigQName

MQSeries initiation queue name. Mandatory parameter. Defines the name of the global MQIA initiation queue.

MRVName

MQIA application name. Mandatory parameter. Used to identify MQIA as a MERVA application. If it is omitted the default value **CMEMQAT** is used.

MRVPwd

MERVA password. Optional if MQIA is started as a batch job. Mandatory if MQIA is started as a Windows NT service. To encrypt the readable password, use the encryption program **cmenypec.exe** that is provided by MQIA.

MRVUser

MERVA user name. Mandatory if used as Windows NT service. Optional if used in a batch job, because a pop-up window asks you for the user name.

MVPIIRecovery

Enable/disable polling mode of the MV2MQ process for all links. Optional parameter. The default is **NO**. Valid values are **YES** and **NO**.

MVQName

MQIA remote user name. Optional parameter. Used for authentication on the remote system by MQSeries. Default value for each link (parameter RmtUser).

ScanInterval

Time-out interval. Optional parameter. The default is 20 seconds. The range is between 10 and 3600 seconds.

Report

Optional parameter. Specifies whether MQIA writes configuration data to the log file, or not. Valid values are **YES** and **NO**. The default is **NO**.

Link-Specific Section Parameters

The section name can be any unique name, for example **[MVQLink1]**.

You can specify the following link-specific parameters:

ConvExitName

Conversion exit name. Optional parameter. Defines the name of an MQSeries conversion exit to be used at the remote site to convert the message. This value is inserted into the message descriptor format field for all messages that are not string messages. The name length is limited to eight characters. If this key is not defined, **MQFMT_NONE** is used.

Note: This parameter is only valid if the remote system provides a conversion exit at the receive channel.

LkName

Link name. Obsolete. The link-specific section header is used as the link name and can be overwritten by **LkName**.

RmtUser

> User name for the remote system. Optional parameter. Overrides the MQIA remote user name **MVQName** in the common section.

MQPIIRecovery

> Enable and disable phase II of the M2QMV recovery processing for the specified link. Optional parameter. The default is **NO**. Valid values are **YES** and **NO**. If not specified, the common section value is used.

MQRcvQName

> MQSeries receive queue. Mandatory parameter.

MQSendQName

> MQSeries send queue. Mandatory parameter.

MVPIIRecovery

> Enable and disable phase II of the MV2MQ recovery processing for the specified link. Optional parameter. The default is **NO**. Valid values are **YES** and **NO**. If not specified, the common section value is used

MVRcvQName

> MERVA receive queue. Mandatory parameter. Can be defined with the following parameters:
>
> * Queue name (mandatory)
> * Queue type (optional)
> * Checking (optional)
>
> **Queue Name**
>
> > Name of the MERVA receive queue.
>
> **Queue Type**
>
> > MQIA recognizes MERVA receive queue types for requests or for datagrams. If the receive queue type is omitted, or only one receive queue is defined for a link, this queue is used for both request and datagram messages. In this case it is impossible for MERVA to distinguish between these message types, unless the message type is defined within the MERVA-related data, for example, in the message comment field. The queue type parameter is appended to the queue name, separated by a comma. The following values are allowed:
> >
> > * 1 = Receive queue is a datagram receive queue
> > * 2 = Receive queue is a request receive queue
> > * No queue type value specifies that it is a common receive queue for both message types
> >
> > If more than one queue of the same receive queue type is specified, MQIA terminates with the appropriate error.
>
> **Checking**
>
> > An additional parameter can be specified for the MERVA receive queue. This parameter activates a check of all incoming SWIFT messages according to SWIFT message rules, and takes place before these messages are put in the MERVA receive queue. The values are:
> >
> > * Y = checking activated
> > * N = checking disabled

If checking is activated and MQIA runs as a Windows NT service, make sure that the user account assigned to the Windows NT service contains a valid MERVA user name, for example **merva1**.

Examples:

The following example defines the datagram (**1**) receive queue **RCV.FROM.RMT**. All incoming SWIFT messages are checked (**Y**):

`RCV.FROM.RMT,1,Y`

The following example defines the receive queue **RCV.FROM.RMT**. The queue is used for both, request and datagram messages. No checking is performed:

`RCV.FROM.RMT,N`

The following example is not valid:

`RCV.FROM.RMT,Y,2`

MVSendQName

MERVA send queue. Mandatory parameter. The MERVA send queue is defined with several additional, optional, and mandatory parameters. Parameters are separated by a comma, and must be in the following order:

Queue name, semaphore name, report option, message priority, field option

Queue Name

Mandatory. Name of the MERVA send queue. The name must be unique for all MERVA queues within MQIA.

Semaphore Name

Mandatory. Must be assigned to the MERVA send queue within the MERVA customization.

Report Options

Mandatory. The report options are specified by a bit-oriented integer value which has the following encoding:

- Bit 0: COA
- Bit 1: COD
- Bit 2: Request message
- Bit 3: Exception report

The report options are ignored for reply messages.

Example: A report option value of 3 (bit 0 = 1, and bit 1 = 1) defines the queue for datagram messages, which require a COA, and a COD from the remote site.

Message Priority

Mandatory. The priority range is MQSeries specific, and is from 0 to 9. It is only used if the MQSeries send queue delivery sequence is set to priority.

Field Options

Optional. This is an additional definition for MERVA send queues. It specifies which MERVA-related field, and if the application

message, must be sent with a message from the MERVA send queue. The parameter is a bit-oriented integer value, and has the following encoding:

- Bit 0: MSG
- Bit 1: MSGOK
- Bit 2: MSGACK
- Bit 3: MSGROUTE
- Bit 4: MSGCMNT
- Bit 5: MSGMAC
- Bit 6: MSGPAC
- Bit 7: TXHEAD
- Bit 8: TXINFO

If this parameter is omitted, the default value is used. The default value is all bits set (= 511), and conforms to the previous version of MQIA.

Note: The application message (bit 0 MSG) is always sent with datagrams and requests, regardless of whether bit 0 of the field options is set. It is only possible to disable the sending of application messages for replies.

Examples:

MSGEXP,ALARMEXP,0,0

This example defines the send queue **MSGEXP** for datagrams that do not require reports, and with a sending priority of 0 (lowest). All valid fields and the application message are sent (default).

MSGEXP,ALARMEXP,0,7

This example defines the send queue **MSGEXP** for datagrams that do not require reports, and with a sending priority of 7. All valid fields and the application message are sent (default).

MSGEXP,ALARMEXP,3,0,12

This example defines the send queue **MSGEXP** for datagrams that require reports COA and COD (3), and with a sending priority of 0 (lowest). For datagrams only, the fields **MSGACK** and **MSGROUTE** (12) are sent with the application message. For replies only, the fields **MSGACK** and **MSGROUTE** are sent without the application message.

MVWaitQName
MERVA wait queue. Mandatory parameter. Can be defined with an optional parameter which can activate an incoming SWIFT message check. This check is according to the SWIFT messaging rules, and takes place before the messages are put in the MERVA wait queue. The values are:

- Y = checking activated
- N = checking disabled

This message check is time consuming.

Examples:

The following example defines the MERVA wait queue **WAIT**. All incoming SWIFT messages are checked (**Y**).

```
WAIT,Y
```

The following example defines the MERVA wait queue **WAIT**. No incoming messages are checked.

```
WAIT
```

Creating the MQIA Desktop Program Object

To invoke MQIA using an MQIA desktop program object, you must create such a desktop program object.

To do this:

1. On the Windows NT desktop, create a new shortcut.
2. In the Create Shortcut window, type the following path and file name in the Command Line field:

   ```
   x:\MERVA-BASE\USE_BRANCH\BIN\CMEMQAT.EXE
   ```

   where **x:\MERVA-BASE** denotes the directory in which MERVA is installed.
3. Append in the Command Line field your desired command line options, see "Command Line Options" on page 45. For example:

   ```
   -f cmemqat.atn -D 3
   ```
4. Click the **Next** button.
5. Change the shortcut description if required.
6. Click the **Finish** button.

How to Migrate from a Previous MQIA Configuration File

The MQIA configuration file in the current version is different to the file in previous versions. The MQIA migration utility enables you to migrate the existing configuration file to the new layout. To do this, execute the following command:

```
rexx cmenymig <old configuration file>
```

Where **<old configuration file>** is the name of the existing file without the file extension. The new file that is created will have the same name, and the file extension of **.ATN**.

If a file with the new configuration file name already exists, the migration utility terminates with an error message. It does not overwrite an existing file.

The migration utility transforms the section headings from the old to the new syntax. The new syntax conforms to the Windows .INI file syntax. It appends a sequential number to the heading of each link-specific section to ensure that the section headings within the file are all unique.

Chapter 7. Channel Message Exit

With MQIA, a channel message exit sample is provided to convert MQSeries message format to and from remote systems for example, from ASCII to EBCDIC. How these channels are defined is described in "Customizing the Channel Message Exit".

The channel message exit sample supports conversion for messages sent to or received from an OS/2, AIX, Windows NT, or MVS system. The supported code pages are listed in "Parameters".

Installing the Channel Message Exit

The channel message exit is installed automatically when you install MERVA. It consists of the following files:

MVQTRXIT.DLL
> This is the channel message exit program used by MQSeries.

MVQTRXIT.C
> This is the source file. If you change this file, set the parameter **DO_DEBUG=1**. You then get the trace file **MVQTRXIT.TRC**, which helps you with coding. After you finish coding, set the parameter **DO_DEBUG=0**.

MVQTRXIT.MAK
> This is the make file for the channel message exit.

The files are located in the following directory:

MERVA base directory\USE_BRANCH\SAMPLES\MQI_ATTACHMENT

Customizing the Channel Message Exit

To assign the channel message exit to MQSeries, define the **MSGEXIT** parameter in the MQSeries configuration. Add the following line in the **MQCONF.MQ** sample file:

```
MSGEXIT('<DIR>\MVQTRXIT (MQAExit)')
```

where **<DIR>** is the directory in which the file **MVQTRXIT.DLL** is located.

To assign the channel message exit to the send channel, add the following line in the **MQCONF.MQ** sample file:

```
MSGDATA (MVS500)
```

This ensures that the remote system provides the parameters required at the send time of a message. At the receive time of the message, MQSeries provides the parameters required. For a detailed description of this parameter, refer to "Parameters".

Parameters

This section describes the **MSGDATA** and the **MSGEXIT** parameters.

MSGDATA Parameter

The **MSGDATA** parameter defines the remote system in terms of decoding and number format. The remote system defines the number format that is to be used:

- MVS and AIX use normal format
- OS/2 and Windows NT use reversed format

The normal format takes the most significant byte first; the reversed format takes the least significant byte first.

The following list shows the values that are valid for the **MSGDATA** parameter. The first three characters define the operating system; the last three characters define the remote coded character set ID (CCSID). The valid CCSID for ASCII is 850 (AIX, OS/2, and Windows NT). Other values, such as 437 and 819 are equivalent to 850. The valid CCSID for EBCDIC is 500 (MVS). The values are case sensitive.

OS2850
> The remote system is OS/2. OS/2 uses ASCII 850 or 437 and wants to receive the message in this format. The number and data formats do not have to be changed.

NT_850
> The remote system is Windows NT. Windows NT uses ASCII 850 or 437 and wants to receive the message in this format. The number and data formats do not have to be changed.

AIX850
> The remote system is AIX. AIX uses ASCII 850 or 819 and wants to receive the message in this format. Only the number format must be changed.

MVS500
> The remote system is an MVS. MVS uses EBCDIC and wants to receive the message in this format. The number and data formats must be changed.

MSGEXIT Parameter

This parameter defines the fully qualified name of the library.

For detailed information on channels and message format, refer to *MQSeries Intercommunication*.

MVQTRXIT Return Codes

The channel message exit handles errors by setting the exit response field in the MQCXP structure to **MQXCC_SUPPRESS_FUNCTION**, and the feedback field to one of the following explanatory codes:

Table 3. MVQTRXIT Return Codes

Code	Description
65540	The version of the **MQCXP** structure is lower than 2
65541	The version of the **MQCD** structure is lower than 3
65542	The exit is not implemented as a channel message exit
65543	The channel type is not valid (not one of the following: SEND, RECEIVE, REQUESTOR, or SERVER)
65544	Send channel, unknown remote system (not AIX, OS2, NT, or MVS)

Table 3. MVQTRXIT Return Codes (continued)

65545	Send channel, message format error for message format **MQFMT_NONE** (calculated length <> given length)
65546	Send channel, remote system MVS, unkown message format (neither **MQFMT_STRING** nor **MQFMT_NONE**, a conversion exit name on MVS)
65547	Send channel, unknown **MQMD** format
65548	Receive channel, unknown **MQMD** format
65549	Receive channel, message format **MQFMT_NONE**, verify fails after integer conversion
65550	Send channel, remote system AIX, unknown message format (either **MQFMT_STRING** or **MQFMT_NONE**)
65551	Unsupported operating system in channel message exit definition (**MSGDATA** field)
65552	Unsupported CCSID in channel message exit definition (**MSGDATA** field)

Using a Conversion Exit Instead of a Channel Message Exit

If a remote system, for example, MERVA ESA, provides an MQSeries conversion exit, you do not have to define the channel message exit on Windows NT. To identify, and use a conversion exit on a remote system, you must specify your conversion exit name in the parameter **ConvExitName** of the MQIA configuration file. This is recommended for a connection between MERVA and MERVA ESA.

For example, use **ConvExitName** = **DSLKCDCC** if MERVA ESA is running under CICS, or **ConvExitName** = **DSLKCDCM** if MERVA ESA is running under IMS. For more information, see the *MERVA for ESA Customization Guide*.

Part 3. Working with MQIA

This part of the book is intended especially for message processing personnel. It describes in detail how to start and work with MQIA.

Chapter 8. Starting and Stopping MQIA

You can start MQIA in the following ways:

- Start the MQIA service (if MQIA is installed as Windows NT service). See "MQIA as Service" on page 46. If the MQIA service is configured to start automatically during operating system startup, MQIA is started automatically.
- From the command line as a batch job (see "Starting and Stopping MQIA as a Batch Job" on page 44).
- Double clicking the MQIA program icon on the desktop.

Prerequisites

To run MQIA, the following applications must be started:

MQSeries queue manager
> This is started by MQIA itself.

MERVA in multi-user mode
> This is started by MQIA, if MQIA is installed as a Windows NT service. If it is not, a user action is required.

If one of the applications has not started, or has failed, do the following:

1. Start MERVA.

   a. In **Programs**, select MERVA USE & Branch.

   b. Select **Control Center**.

   c. Double-click **Local System**.

   d. Select the MERVA instance.

   e. Right-click the selected MERVA instance.

   f. Click **Start in multi user mode**.

   For more detailed information about starting MERVA, refer to the *MERVA USE & Branch for Windows NT User's Guide*.

2. Start the MQSeries server if MQIA uses an MQSeries server connection. You can do this within MQSeries explorer, or in a command prompt window by entering:

   `strmqm QMGR1`

   The message **MQSeries queue manager started** is displayed. If you do not get this message, or if you get an error message, check the Windows NT event log and the MQSeries error logs (see "MQIA Logging" on page 63). If MQIA is using a client connection, verify that the server connection channel is set up on the MQSeries server, and that the client connection channel is defined on the local machine either by a client channel table, or by the appropriate environment variables (see *MQSeries Client Guide*).

   Note: If the queue manager is not started, and if MQIA uses an MQSeries server connection, MQIA tries to start the queue manager.

3. Verify that the MQSeries listener is running.

   Usually, the MQSeries listener is started automatically when you start Windows NT. If not, open a command prompt and enter the following commands:

```
RUNMQSC QMGR1
START LISTENER
```

You then get the message **MQSeries Listener started**. If you do not get this message, or if you get an error message, check the MQSeries error logs for the appropriate queue manager, see "MQIA Logging" on page 63.

4. Verify that the MQSeries send channel is running.

   Usually, the send channels are started automatically when the queue manager is started. To check that the send channel is started, do the following:

   a. Open a command prompt window, and enter the following commands:
   ```
   DIS CHS (*)
   ```

   This displays the current channel status of all active channels.

   b. If the MQSeries send channel is not started, enter the following command:
   ```
   STA CHL (SND.TO.REMOTE)
   ```

   where **SND.TO.REMOTE** denotes the name of the send channel.

   c. If the MQSeries send channel is not started, do the following:

   1) Check the MQSeries error log **AMQERR01.log** that is located in the following directory:

      <MQSeries base directory>\QMGRS\<QUEUE MANAGER>\ERRORS

      where **<QUEUE MANAGER>** denotes the name of the queue manager.

   2) If the message counter is different to that of the send channel on the remote side, reset the send channel so that the message counters of the send channel match the message counters of the receive channel. To do this, enter the following command:
   ```
   RESET CHL (SND.TO.REMOTE)
   ```

   For more detailed information, refer to the *MQSeries Intercommunication Guide*.

5. Verify that the MQSeries receive channel and the send channels of the remote site are running.

   Usually, the receive channels are enabled automatically when MQSeries is started. The receive channel is started when the remote site starts its send channels.

   • If the status is **Not Running** or **Retry**, check the MQSeries error log **AMQERR01.log**. This file is located in the directory:

      <MQSeries base directory>\QMGRS\<QUEUE MANAGER>\ERRORS

   • If the status is **Binding**, starting of the channel is not yet complete. Wait a short period of time, then check the status again.

After these conditions are met, you can start MQIA.

Starting and Stopping MQIA as a Batch Job

To start MQIA, double-click the **MQIA** icon, or enter the following command in a Command Prompt window:
```
CMEMQAT
```

Command Line Options

When you start MQIA, you can specify different options. The following section shows the syntax and a description of the parameters.

The syntax is:
```
CMEMQAT
[-f <configuration file name>]
[-I]
[-D <log level>]
[-U <MERVA username>]
[-P <MERVA password>]
[-S <Process with queue polling>]
```

For example:
```
CMEMQAT -f MYCFG -I -D 1 -U JOHN -P MARY -S 2
```

The following list describes the command line parameters:

-f   This parameter defines the name of the file that contains the MQIA configuration. The file extension can be omitted because it is always extended to **.ATN**. If this parameter is omitted, a default file name is used, which is the same as the current MERVA instance name. For example, if the current MERVA instance is **MERVA1**, the configuration file name is **MERVA1.ATN**.

-I   This parameter defines whether a summary of the configuration file is written to the MQIA log file. The default is off.

-D   This parameter defines the level of detail for logging. The following MQIA log levels are possible:
- Error messages only (Level 1)
- Error and warning messages (Level 2)
- Errors, warning, and information messages (Level 3)
- Any type of log message is logged (very time consuming, and uses lots of disk space) (Level 4)

     The default is 1.
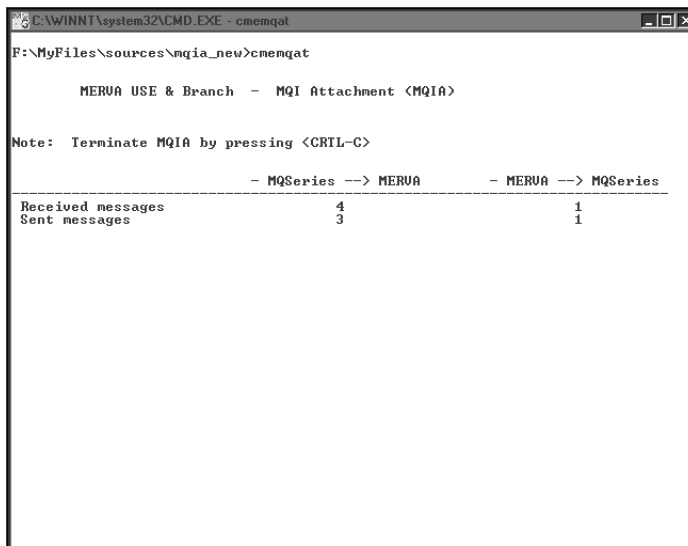
-U   The MERVA user name. If it is not specified, and not defined in the MQIA configuration file, a dialog pops up during MQIA startup, and you are asked to type in a MERVA user name.

-P   The MERVA password. If it is not specified and is not defined in the MQIA configuration file, a dialog pops up during MQIA startup, and you are asked to enter a MERVA password.

-S   Defines the process that periodically polls its input queues. The polling interval is defined in the MQIA profile. The possible values are:
- 0: No polling is done for both processes
- 1: The MQIA process periodically scans its MQSeries receive queues
- 2: The MERVA process periodically scans its MERVA send queues
- 3: The MQIA process and the MERVA process poll their input queues (MQSeries receive queues for MQIA, and MERVA send queues for the MERVA process)

     This polling activity recovers all the messages that are not processed during normal processing, for example, locked messages in the MERVA queue.

The default is 0.

The MQIA Window

When you start MQIA in a command prompt window, the MQI Attachment window is shown.


```
MQ  C:\WINNT\system32\CMD.EXE - cmemqat                                    _ □ ×

F:\MyFiles\sources\mqia_new>cmemqat

        MERVA USE & Branch  -  MQI Attachment (MQIA)


Note:  Terminate MQIA by pressing (CRTL-C)

                              - MQSeries --> MERVA      - MERVA --> MQSeries
-----------------------------------------------------------------------------
Received messages                       4                       1
Sent messages                           3                       1
```

Figure 15. MQI Attachment Window

The MQI Attachment window displays the statistics about received messages, and successfully processed (sent) messages for both MQIA processes. Successfully processed messages are those messages that are processed without any errors, and are put in the appropriate target queue, which is:

- The MERVA receive queue for the MQ2MV process (MQSeries —> MERVA column)
- The MQSeries send queue for the MV2MQ process (MERVA —> MQSeries column)

The MQIA screen also shows if a process is in normal processing mode, or in recovery mode. Normal processing mode is indicated by a dash (-) before the table headings, as shown in Figure 15 for both processes. An exclamation point (!) indicates recovery processing.

To end MQIA, press **CTRL-C**, then confirm that you want to terminate MQIA.

MQIA as Service

MQIA can be installed and run as Windows NT service. This means that MQIA can be started when Windows NT starts up, and run in the background. MQIA service administration is provided within the MERVA Control Center.

Installing MQIA Service

To install an MQIA service:

1. Start the MERVA Control Center. To do this, select **Start -> Programs -> MERVA USE & Branch -> Control Center**. The MERVA Control Center window is displayed, see Figure 16 on page 47.
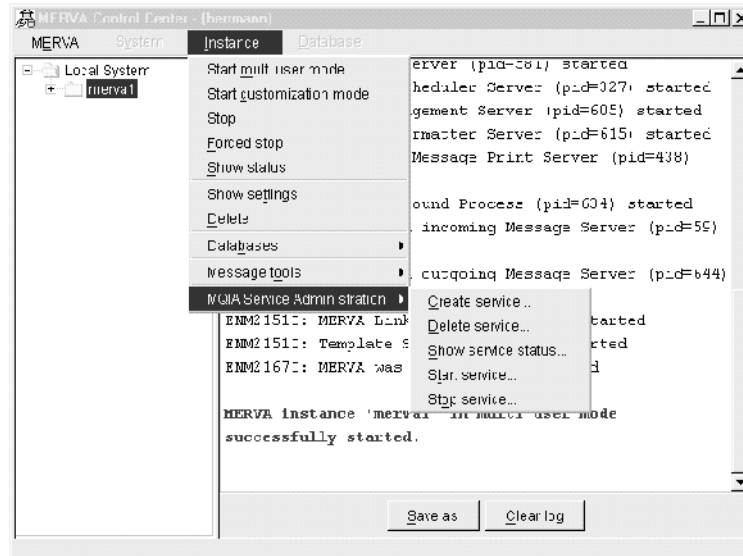
Figure 16. Installing an MQIA Service Instance

2. Select the MERVA instance in the tree, and open the instance menu.
3. Select the menu item **MQIA Service Administration**.
4. Select the menu item **Create service...**. A dialog box pops up, as shown in Figure 17.
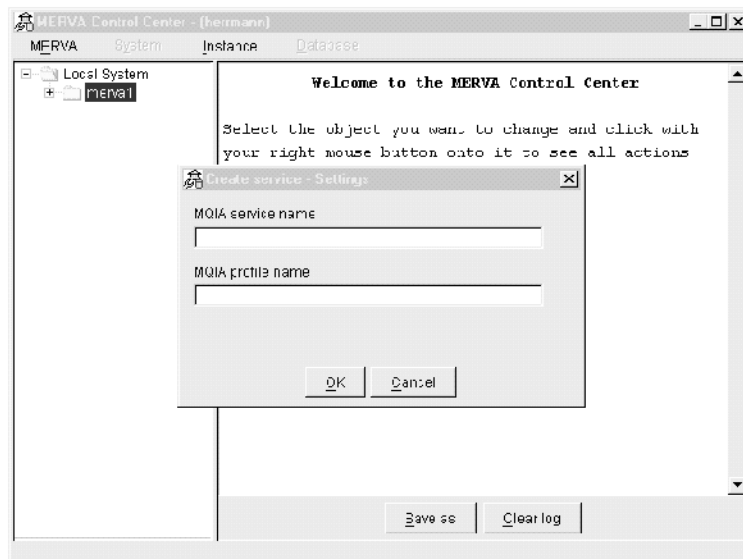
Figure 17. Creating MQIA Service

5. Enter the service name. The specified service name is also used within the display name of the service in the services applet. The service names are constructed as follows:

```
MERVA - MQIA -<service name>
```

Where **<service name>** is replaced by the value that you enter in the MQIA service name field in the **Create service - Settings** dialog.

6. If you are not using the default profile, enter a valid profile name (configuration file).

   The profile name must be a fully qualified file name. If you are using the default profile, you can leave the appropriate entry field empty.

7. Click **OK** to create the service. Any errors are displayed in the right window of the MERVA Control Center.

Starting MQIA Service

To start MQIA service:

1. Start the MERVA Control Center. To do this select the MERVA Control Center menu item from the MERVA USE & Branch start menu.
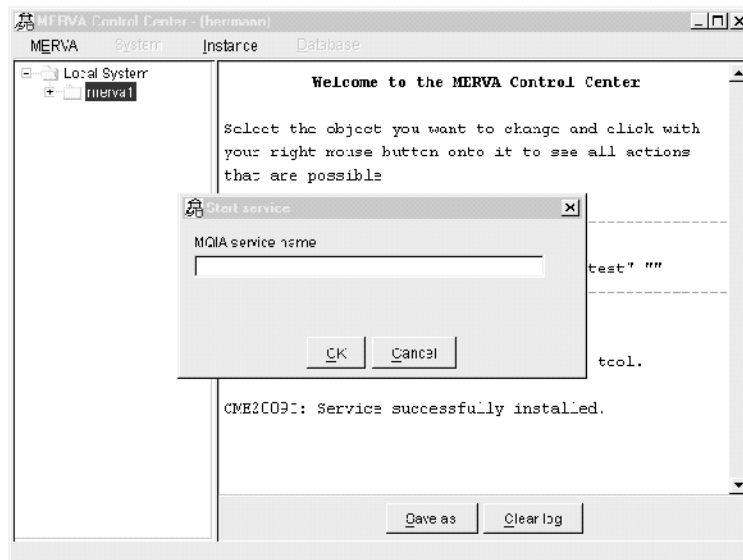
Figure 18. Starting MQIA Service

2. Select the MERVA instance in the tree, and open the instance menu.
3. Select the menu item **MQIA Service Administration**.
4. Select the menu item **Start service...**. A dialog box pops up, as shown in Figure 18.
5. Enter a valid MQIA service name. For information about installed MQIA services, you can use the Windows NT services applet.
6. Click **OK** to start the service.

   A successful service start message does not mean that the service is running. Any error that occurs during MQIA start up can cause the service to terminate. To verify if the service is running, use the **Show service status...** menu item, see "Displaying the MQIA Service Status".

   To find out why the service has terminated, see the Windows NT event log and the MQIA log file (see "MQIA Logging" on page 63).

Displaying the MQIA Service Status

To check if the MQIA service is running:

1. Start the MERVA Control Center. To do this, select the MERVA Control Center menu item from the MERVA USE & Branch start menu.
2. Select the MERVA instance in the tree, and open the instance menu.

3. Select the menu item **MQIA Service Administration**.

4. Select the menu item **Show service status...**. A dialog box pops up.

5. Enter a valid MQIA service name. For information about installed MQIA services, you can use the Windows NT services applet.

6. Click **OK** to display the specified MQIA service status. The status is displayed in the right-hand window of the MERVA Control Center.

Stopping MQIA Service

To stop MQIA service:

1. Start MERVA Control Center. To do this, select the MERVA Control Center menu item from the MERVA USE & Branch start menu.

2. Select the MERVA instance in the tree, and open the instance menu.

3. Select the menu item **MQIA Service Administration**.

4. Select the menu item **Stop service...**. A dialog box pops up.

5. Enter a valid MQIA service name. For information about installed MQIA services, you can use the Windows NT services applet.

6. Click **OK** to stop the specified MQIA service. The result of the stop command is displayed in the right-hand window of the MERVA Control Center.

Deleting MQIA Service

To delete MQIA service:

1. Start MERVA Control Center. To do this, select the MERVA Control Center menu item from the MERVA USE & Branch start menu.

2. Select the MERVA instance in the tree, and open the instance menu.

3. Select the menu item **MQIA Service Administration**.

4. Select the menu item **Delete service...**. A dialog box pops up.

5. Enter a valid MQIA service name. For information about installed MQIA services, you can use the Windows NT services applet.

6. Click **OK** to delete the specified MQIA service. The result of the delete command is displayed in the right-hand window of the MERVA Control Center.

Part 4. Appendixes

Appendix A. Definition Sample of MQCONF.MQ

The file, **MQCONF.MQ**, is a sample file that defines all MQSeries objects needed by MQIA. For more information see *MQSeries Command Reference*.

```
*------------------------------------------------------------------------------*
*
*   Definition File for queue manager QMGR1
*
*------------------------------------------------------------------------------*


*******************************************************************************
* >>>> COMMON RESET SECTION
*
DELETE QLOCAL(DLQ)          PURGE
DELETE QLOCAL(DXQ)           PURGE
DELETE QLOCAL(REFQ)          PURGE
DELETE QLOCAL(TRIGQ)         PURGE
DELETE QLOCAL(TACTQ)         PURGE
DELETE PROCESS(ENMMQAT)
*
* END COMMON RESET SECTION
*******************************************************************************


*******************************************************************************
* >>>> LINK SPECIFIC RESET SECTION    for Link No. 01 <<<<
*
*
STOP    CHANNEL(REMOTE.TO.LOCAL)      MODE(FORCE)
DELETE CHANNEL(REMOTE.TO.LOCAL)
STOP    CHANNEL(LOCAL.TO.REMOTE)      MODE(FORCE)
DELETE CHANNEL(LOCAL.TO.REMOTE)
DELETE QREMOTE(SND.TO.REMOTE)
DELETE QLOCAL(REMOTE)                 PURGE
DELETE QLOCAL(RCV.FROM.REMOTE)         PURGE
*
* END LINK SPECIFIC RESET SECTION
*******************************************************************************


*******************************************************************************
* >>>> COMMON DEFINITION SECTION <<<<
* common elements.
*
DEFINE QLOCAL(DLQ)                                          +
       REPLACE                                             +
       DEFPRTY(0)                                          +
       DEFPSIST(YES)                                       +
       DESCR('Local Dead-letter queue')                    +
       PUT(ENABLED)                                        +
       DEFSOPT(SHARED)                                     +
       GET(ENABLED)                                        +
       MAXDEPTH(99999)                                     +
       MAXMSGL(31000)                                      +
       MSGDLVSQ(PRIORITY)                                  +
       SHARE                                               +
       NOTRIGGER                                           +
       QDEPTHHI(0)                                         +
       QDEPTHLO(100)                                       +
       QDPHIEV(DISABLED)                                   +
       QDPLOEV(DISABLED)                                   +
       QDPMAXEV(DISABLED)                                  +
       QSVCIEV(NONE)                                       +
```

```
                         SCOPE(QMGR)                              +
                         USAGE(NORMAL)

            DEFINE QLOCAL(REFQ)                                   +
                         REPLACE                                 +
                         DEFPRTY(0)                              +
                         DEFPSIST(YES)                           +
                         DESCR('Local Reference queue')          +
                         PUT(ENABLED)                            +
                         DEFSOPT(SHARED)                         +
                         GET(ENABLED)                            +
                         MAXDEPTH(99999)                         +
                         MAXMSGL(31000)                          +
                         MSGDLVSQ(PRIORITY)                      +
                         SHARE                                   +
                         NOTRIGGER                               +
                         QDEPTHHI(0)                             +
                         QDEPTHLO(100)                           +
                         QDPHIEV(DISABLED)                       +
                         QDPLOEV(DISABLED)                       +
                         QDPMAXEV(DISABLED)                      +
                         QSVCIEV(NONE)                           +
                         SCOPE(QMGR)                             +
                         USAGE(NORMAL)

            DEFINE QLOCAL(TACTQ)                                 +
                         REPLACE                                 +
                         DEFPRTY(0)                              +
                         DEFPSIST(YES)                           +
                         DESCR('Local Transaction queue')        +
                         PUT(ENABLED)                            +
                         DEFSOPT(SHARED)                         +
                         GET(ENABLED)                            +
                         MAXDEPTH(99999)                         +
                         MAXMSGL(31000)                          +
                         MSGDLVSQ(PRIORITY)                      +
                         SHARE                                   +
                         NOTRIGGER                               +
                         QDEPTHHI(0)                             +
                         QDEPTHLO(100)                           +
                         QDPHIEV(DISABLED)                       +
                         QDPLOEV(DISABLED)                       +
                         QDPMAXEV(DISABLED)                      +
                         QSVCIEV(NONE)                           +
                         SCOPE(QMGR)                             +
                         USAGE(NORMAL)


            DEFINE QLOCAL(TRIGQ)                                 +
                         REPLACE                                 +
                         DEFPRTY(0)                              +
                         DEFPSIST(NO)                            +
                         DESCR('Local initiation queue')         +
                         PUT(ENABLED)                            +
                         DEFSOPT(SHARED)                         +
                         GET(ENABLED)                            +
                         MAXDEPTH(99999)                         +
                         MAXMSGL(1000)                           +
                         MSGDLVSQ(PRIORITY)                      +
                         SHARE                                   +
                         NOTRIGGER                               +
                         QDEPTHHI(0)                             +
                         QDEPTHLO(100)                           +
                         QDPHIEV(DISABLED)                       +
                         QDPLOEV(DISABLED)                       +
                         QDPMAXEV(DISABLED)                      +
                         QSVCIEV(NONE)                           +
```

```
          SCOPE(QMGR)                                         +
          USAGE(NORMAL)

DEFINE QLOCAL(DXQ)                                            +
          REPLACE                                             +
          DEFPRTY(0)                                          +
          DEFPSIST(YES)                                       +
          DESCR('Default transmission queue')                +
          PUT(ENABLED)                                        +
          DEFSOPT(SHARED)                                     +
          GET(ENABLED)                                        +
          MAXDEPTH(99999)                                     +
          MAXMSGL(31000)                                      +
          MSGDLVSQ(PRIORITY)                                  +
          SHARE                                               +
          NOTRIGGER                                           +
          QDEPTHHI(0)                                         +
          QDEPTHLO(100)                                       +
          QDPHIEV(DISABLED)                                   +
          QDPLOEV(DISABLED)                                   +
          QDPMAXEV(DISABLED)                                  +
          QSVCIEV(NONE)                                       +
          SCOPE(QMGR)                                         +
          USAGE(XMITQ)
*
 DEFINE PROCESS(ENMMQAT)                                      +
          REPLACE                                             +
          APPLTYPE(WindowsNT)                                 +
          DESCR('ENMMQAT application: CMEMQAT')
*
*
* END COMMON DEFINITION SECTION
********************************************************************************

********************************************************************************
*  >>>> LINK SPECIFIC DEFINITION SECTION    for Link No. 01  <<<<
* Used to connect QMGR1 with REMOTE.
*
DEFINE CHANNEL(REMOTE.TO.LOCAL)                               +
          CHLTYPE(RCVR)                                       +
          TRPTYPE(TCP)                                        +
          DESCR('RCV Channel from REMOTE')                    +
          REPLACE                                             +
          MAXMSGL(32000)                                      +
* +++ User definition follows : message channel exit
* +++ and remote system type ...
*         MSGEXIT('MVQTRXIT(MQAExit)')                        +
*         MSGDATA(MVS500)                                     +
* --- End of User definition.
          SEQWRAP(255)                                        +
          MRRTY(3)                                            +
          MRTMR(30)

DEFINE CHANNEL(LOCAL.TO.REMOTE)                               +
          CHLTYPE(SDR)                                        +
* +++ User definition follows : remote partners TCP/IP address ...
          CONNAME('1.255.244.8(1414)')                        +
* --- End of User definition.
          TRPTYPE(TCP)                                        +
* +++ User definition follows :
*    local transmit q ( named as remote queue manager ! )
          XMITQ(REMOTE)                                       +
* --- End of User definition.
          CONVERT(NO)                                         +
          DESCR('Send CHANNEL to REMOTE')                     +
          REPLACE                                             +
          DISCINT(0)                                          +
```

```
                LONGRTY(999999)                                        +
                LONGTMR(30)                                            +
                MAXMSGL(32000)                                         +
* +++ User definition follows : message channel exit and remote system type ...
*          MSGEXIT('MVQTRXIT(MQAExit)')                               +
*          MSGDATA(MVS500)                                            +
* --- End of User definition.
                SEQWRAP(255)                                          +
                SHORTRTY(3)                                           +
                SHORTTMR(3)

 DEFINE QLOCAL(RCV.FROM.REMOTE)                                       +
                REPLACE                                               +
                DEFPRTY(0)                                            +
                DEFPSIST(YES)                                         +
                DESCR('Receive Queue from REMOTE')                    +
                PUT(ENABLED)                                          +
                DEFSOPT(SHARED)                                       +
                GET(ENABLED)                                          +
                INITQ(TRIGQ)                                          +
                MAXDEPTH(99999)                                       +
                MAXMSGL(32000)                                        +
                MSGDLVSQ(PRIORITY)                                    +
                SHARE                                                 +
                TRIGGER                                               +
                PROCESS(ENMMQAT)                                      +
                QDEPTHHI(0)                                           +
                QDEPTHLO(100)                                         +
                QDPHIEV(DISABLED)                                     +
                QDPLOEV(DISABLED)                                     +
                QDPMAXEV(DISABLED)                                    +
                QSVCIEV(NONE)                                         +
                SCOPE(QMGR)                                           +
                TRIGDATA('RCV from remote system')                   +
                TRIGDPTH(1)                                           +
                TRIGMPRI(0)                                           +
                TRIGTYPE(EVERY)                                       +
                USAGE(NORMAL)


    * +++ User definition follows :
    *     this local transmission queue must be named like
    *     the remote queue manager !
 DEFINE QLOCAL(REMOTE)                                                +
 * --- End of User definition.
                REPLACE                                               +
                DEFPRTY(0)                                            +
                DEFPSIST(YES)                                         +
                DESCR('Local transmit queue to REMOTE')              +
                PUT(ENABLED)                                          +
                DEFSOPT(SHARED)                                       +
                GET(ENABLED)                                          +
                MAXDEPTH(99999)                                       +
                MAXMSGL(31000)                                        +
                MSGDLVSQ(PRIORITY)                                    +
                INITQ(SYSTEM.CHANNEL.INITQ)                          +
                SHARE                                                 +
 * +++ Enable triggering on transmission queue
 * +++ Start send channel on arrival of first message
                TRIGGER                                               +
                QDEPTHHI(0)                                           +
                QDEPTHLO(100)                                         +
                QDPHIEV(DISABLED)                                     +
                QDPLOEV(DISABLED)                                     +
                QDPMAXEV(DISABLED)                                    +
                QSVCIEV(NONE)                                         +
                SCOPE(QMGR)                                           +
```

```
                TRIGDATA('LOCAL.TO.REMOTE')                          +
                TRIGDPTH(1)                                          +
                TRIGMPRI(0)                                          +
                TRIGTYPE(FIRST)                                      +
                USAGE(XMITQ)

DEFINE QREMOTE(SND.TO.REMOTE)                                        +
        REPLACE                                                     +
        DEFPRTY(0)                                                  +
        DEFPSIST(YES)                                               +
        DESCR('Send queue to REMOTE')                              +
        PUT(ENABLED)                                                +
* +++ User definition follows :
*     remote receive queue name,
*     remote queue manager name,
*     local transmission queue name
*     (named like the remote queue manager!)
        RNAME(RCV.FROM.QMGR1)                                       +
        RQMNAME(REMOTE)                                             +
        XMITQ(REMOTE)                                               +
* --- End of User definition.
        SCOPE(QMGR)
*
* END LINK SPECIFIC DEFINITION SECTION    for Link No. 01
*******************************************************************************
```

Appendix B. Sample Configuration File CMEMQAT.ATN

The following sample configuration file, **CMEMQAT.ATN**, contains the common section [MVQAttachment], and one link specific section (MVQLink1).

```
#Configuration File for MQ-Attachment V4.12 for MERVA USE & Branch
# (1) => to be customized in MERVA.
# (2) => to be configured  in MQSeries

# Common Section
[MVQAttachment]
  MVQName = mvqatTP              # Default remote operating system user ID
  MRVName = MervaMQ              # MERVA application ID
  MRVUser = merva1              # (1) MERVA user name
#  MRVPwd  = _____              # (1) MERVA password
  MQQMgrName = QMGR1            # (2) MQSeries queue manager name
  MQRefQName = REFQ            # (2) MQSeries Reference queue name
#  MQTActQName = TACTQ          # (2) MQSeries Transaction queue name
  MQTrigQName = TRIGQ          # (2) MQSeries initiation (Trigger) queue name
  ScanInterval = 60            # Message waiting timeout (10..3600 seconds)
  LogDir = c:\temp            # Logging directory name
#  MQPIIRecovery = Yes          # The MQSeries receive queue polling for all links is on
#  MVPIIRecovery = Yes          # The MERVA send queue polling for all links is on
#  APITrace = No              # The MERVA API trace is off
#  MQClient = No              # The MQSeries server connection is used
#  LogLevel = 3              # Log level is set to 3 -> error, info and warning messages

# First Link Section
[MVQLink1]                    # Section heading
#  LkName = MVQLK1            # Link name. Overwrites section heading
  RmtUser = remote            # (2) Remote Operating System user ID, link-specific
  MQRcvQName  = RCV.FROM.RMT  # (2) MQI receive queue
  MQSendQName = SND.TO.RMT    # (2) MQI send queue
  MVRcvQName  = SL_IN        # (1) MERVA receive queue. Common for requests and datagrams
  MVWaitQName = WAIT        # (1) MERVA wait queue
  MVSendQName = SL_OUT,ALARMEXP,7,0 # (1) MERVA send queue, semaphore, report level, priority
#  ConvExitName = CNV2NT      # Conversion Exit Name used by MQSeries on remote site
#  MQPIIRecovery = No          # The MQSeries receive queue polling for this link is off
#  MVPIIRecovery = No          # The MERVA send queue polling for this link is off
```

For more details see "Chapter 6. Customizing MQIA" on page 29.

Appendix C. Problem Determination

Errors can occur in the following places:

- Local MQSeries queue manager
- Remote MQSeries queue manager
- MERVA
- Windows NT
- MQIA

For each error, an error message is logged in the MQIA log file. The error messages are explained in "Appendix F. Error Messages and Codes" on page 69. Different error code identifiers are used in the log file to identify from where the error originated. These error code identifiers are:

Completion and Reason Code

> Error messages containing completion and reason codes are caused by an MQSeries queue manager. For more details about the error, look up the codes in the *MQSeries Messages* book.

MERVA Error

> Error messages containing MERVA error codes are caused by MERVA. For more details about these errors, see the *MERVA Application Programming Guide*.

OS Error

> Error messages containing OS error codes are caused by the Windows NT operating system. For more details see the *Windows NT SDK*.

MQIA Error

> Error messages containing MQIA error codes are caused by MQIA itself.

Dead-Letter Reason Codes

> If MQIA could not process a message, it is put in the dead-letter queue. The message is expanded by a dead-letter queue header. The field "Reason" in the dead-letter queue header is filled with an appropriate reason code. These reason codes are described in Table 4.

Table 4. Dead-Letter Reason Codes

Dead-Letter Reason Code	Description	Explanation
1	Reserved, currently not used	
2	No corresponding message found on MQSeries reference queue	An incoming report or reply could not be correlated because there was no corresponding request or datagram reference message found on the MQSeries reference queue. Check that the message IDs and MRNs are unique.
3	Request with current message ID already exists	A request with the current message ID already exists in the MQSeries reference queue. Message IDs must be unique.
4	Invalid application (SWIFT or Telex) message format	

Table 4. Dead-Letter Reason Codes (continued)

5	No corresponding message found on the MERVA wait queue	The incoming report or reply could not be correlated to a corresponding message in the MERVA wait queue.
6	MERVA routing error	A message could not be routed to its target queue within MERVA.
7	Invalid MQSeries message type	The message type in the MQSeries message descriptor of the current message is not supported by MQIA.
8	Invalid MQSeries message layout	MQIA does not support the message layout. MQIA supports only the five layouts described in this User's Guide. Check your remote application, or verify if the data conversion (if necessary) on MQSeries is correctly configured.
9	Invalid MERVA-related field value	
10	Application message in received MQSeries message is too big for MERVA	The application message within the MQSeries message is too big for MERVA. The maximum length is 28,000 bytes.
11	Invalid MQSeries trigger message	An invalid message was received in the MQSeries initiation queue used by MQIA.
12	Invalid MQSeries source queue name in MQSeries trigger message	The source queue name in the received trigger message is not valid for MQIA. This can be because another MQSeries receive queue has set the initiation queue name to the wrong initiation queue.

If MQIA receives an exception report from MQSeries, error handling, as described in "Appendix D. Exception Reports Sent by Remote MQSeries Queue Manager" on page 65, takes place.

When you report an MQIA problem, you must provide the following information:

- System Configuration. Use the maintenance utility **ENMNCONF** from the MERVA USE & Branch product to obtain the system information.
- **CMEMQAT** and queue manager configuration files:
  - The file used to set up MQSeries, for example, **MQCONF.MQ**
  - The **CMEMQAT** configuration file
- The MQIA log file
- MERVA diagnosis log
- MERVA programmer's trace
- MERVA customization report
- MQSeries queue manager error log
- MQSeries error log
- Description of any changes to the configuration before the error occurred.

See "MQIA Logging" on page 63 for information on how to find and get these logs.

MQIA Logging

During MQIA processing, errors can occur in any of the components involved. Each component has its own logging strategy, directories, and log files.

MQIA

- MQIA log file. This file is located in the directory specified in the parameter **LogDir** of the MQIA configuration file. The log file has the same name as the MQIA configuration file currently in use, but it has the file extension **.LOG**.
- Windows NT event log. This is only used until the MQIA log file is opened, or if a logging error occurs that the MQIA server cannot write to the MQIA log file.

MERVA

- **ENMAPI.TRC** located in the directory:

  <Logging path>\TRACES\API
- **ENMBASE.TRC** located in the directory:

  <Logging path>\TRACES\BASE
- **ENMDIAG.LOG** located in the directory:

  <Logging path>\LOGS\BASE

  where **<Logging path>** is the directory defined when your created your MERVA instance. For more details, see *MERVA Diagnosis Guide*.

MQSeries
AMQERR01.LOG

MQSeries log files can be found as follows:

- MQSeries queue manager specific log file. This is located in the directory:

  <MQSeries base directory>\qmgrs\<Queue Manager Name>\errors
- MQSeries specific log file if the queue manager is not available. This is located in the directory:

  <MQSeries base directory>\system\errors
- MQSeries client specific log file. This is located in the directory:

  <MQSeries base directory>\errors

  where **<MQSeries base directory>** is the directory where MQSeries is installed. See *MQSeries System Administration Guide*.

Log file entries start with an eyecatcher and a sequence number. If an entry is written into the Windows NT event log, the sequence numbering in the MQIA log file shows a gap.
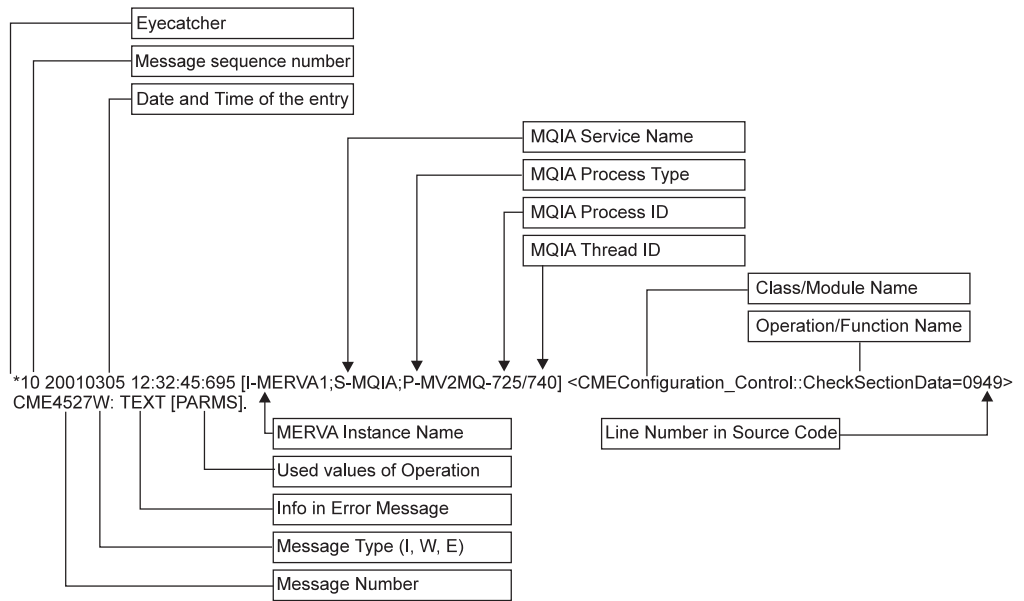
Figure 19. MQIA Log File Layout

The MQIA process type can be either MV2MQ or MQ2MV.

MQIA recognizes the following log message types:

- **E**: Error messages (log level 1, and higher)
- **W**: Warning messages (log level 2, and higher)
- **I**: Information and trace messages (information = log level 3, and higher, trace = log level 4)

The MQIA service name is only valid if MQIA is running as service.

Appendix D. Exception Reports Sent by Remote MQSeries Queue Manager

When MQIA receives an exception report from MQSeries, the **MSGOK** field of the corresponding message in the MERVA wait queue is updated with the appropriate error code as shown in Table 5. Table 5 shows the relationship between the feedback code in the exception report message descriptor, received from the remote site, and the error code in the MERVA message **MSGOK** field.

Table 5. Relationship of MQSeries Feedback Code to MSGOK Field Contents

MQMD Feedback Code	MSGOK Field Error Code
MQFB_TM_ERROR	MRVNOK02
MQFB_APPL_TYPE_ERROR	MRVNOK03
MQRC_PUT_INHIBITED	MRVNOK04
MQRC_Q_FULL	MRVNOK05
MQRC_NOT_AUTHORIZED	MRVNOK06
MQRC_Q_SPACE_NOT_AVAILABLE	MRVNOK07
MQRC_MSG_TOO_BIG_FOR_Q_MGR	MRVNOK08
MQRC_PERSISTANT_NOT_ALLOWED	MRVNOK09
Any other feedback code	MRVNOK99

For more information see *MQSeries Application Programming Interface Reference* and *MQSeries Messages*.

Appendix E. MQIA Password Encryption Utility

MQIA provides a password encryption utility which allows you to encrypt the MERVA password defined in the MQIA configuration file.

You invoke the utility by typing the following in a command prompt window:

```
cmenypec <profile name> [<-n>]
```

Where**<profile name>** is an MQIA configuration file including the file extension **.ATN**. If the configuration file is not fully qualified, the current directory is used.

<-n> is an optional argument. If you specify it, the encryption utility asks for a new user name and password.

If no MERVA user name is defined in the configuration file, you are prompted to specify one. If no password is defined in the configuration file, you are prompted to type in a password. The password is then encrypted and both, user name and password values, are written to the configuration file. A password that is already encrypted, is left unchanged.

Appendix F. Error Messages and Codes

CME2000E **The command line arguments are syntactically invalid.**

Explanation: The command line arguments are syntactically invalid.

System Action: The command was not processed.

User Response: Call the program with parameter '-h' to get more information.

CME2001E **Command line arguments are semantically invalid.**

Explanation: The command line arguments are semantically invalid.

System Action: The command was not processed.

User Response: Call the program with parameter '-h' to get more information.

CME2002E **Cannot register event logging.**

Explanation: Could not make the necessary entries in the Windows registry.

System Action: The command was not processed.

User Response: Increase the size of the registry, reboot the system, and reissue the command.

CME2003W **Cannot unregister event logging.**

Explanation: No entries could be deleted from the Windows registry.

System Action: The command was not processed.

User Response: Reboot the system and reissue the command.

CME2004E **Cannot undo register event logging.**

Explanation: No entries could be restored in the Windows registry.

System Action: The command was not processed.

User Response: Reboot the system, remove and reinstall the service.

CME2005E **Cannot get access to service due to a general error.**

Explanation: The service installation and administration tool has failed to get access to the service from the Windows service control manager (SCM).

System Action: The command was not processed.

User Response: Reboot the system. If the error continues, contact your IBM service representative.

CME2006E **Cannot get access to the service control manager (SCM).**

Explanation: The service installation and administration tool cannot get access to the Windows service control manager (SCM).

System Action: The command was not processed.

User Response: Reboot the system. If the error continues, contact your IBM service representative.

CME2007E **Cannot open the service control manager (SCM) database due to a general error.**

Explanation: The service installation and administration tool cannot get access to the Windows service control manager (SCM) database.

System Action: The command was not processed.

User Response: Reboot the system. If the error continues, contact your IBM service representative.

CME2008E **Cannot get access to service due to an invalid service name.**

Explanation: The service control manager (SCM) has rejected the specified service name.

System Action: The command was not processed.

User Response: Repeat the command with a different service name.

CME2009I **Service successfully installed.**

Explanation: The service has been successfully installed to the Windows service control manager (SCM) database.

System Action: None.

User Response: None.

CME2010E **Cannot create service due to a circular service dependency.**

Explanation: The Windows service control manager (SCM) cannot find a valid sequence to start all dependent services.

System Action: The command was not processed.

User Response: Check the specified dependency list and remove the circular dependency.

CME2011E   **Cannot create service due to an invalid display name.**

Explanation: The Windows service control manager (SCM) has rejected the specified service display name.

System Action: The command was not processed.

User Response: Reissue the command with a different service display name.

CME2012E   **Cannot create service due to an invalid parameter.**

Explanation: The Windows service control manager (SCM) has rejected a specified service creation parameter.

System Action: The command was not processed.

User Response: Reissue the command with different parameters.

CME2013E   **Cannot create service due to an invalid user account.**

Explanation: The specified user account does not exist or does not have the Windows system right "Load and unload device drivers" or "Log on as a service".

System Action: The command was not processed.

User Response: Reissue the command using a different user account, or update the corresponding user rights.

CME2014E   **Cannot create service because a service with the same name already exists.**

Explanation: Cannot create a service because a service with the same name already exists.

System Action: The command was not processed.

User Response: Reissue the command with a different service name.

CME2015E   **Cannot create service due to an invalid service name.**

Explanation: The service control manager (SCM) rejected the specified service name.

System Action: The command was not processed.

User Response: Reissue the command with a different service name.

CME2016I   **Service successfully removed.**

Explanation: The service was successfully removed from the Windows service control manager (SCM) database.

System Action: None.

User Response: None.

CME2017E   **Cannot get access to service because service does not exist.**

Explanation: The Windows service control manager (SCM) does not know this service.

System Action: The command was not processed.

User Response: Reissue the command with an existing service name.

CME2018E   **Cannot remove service because service is still running.**

Explanation: The service must be stopped before it can be removed.

System Action: The command was not processed.

User Response: Stop the service and reissue the command.

CME2019W   **Service already marked for removal.**

Explanation: The service was marked for deletion more than once.

System Action: None.

User Response: None.

CME2020I   **Service has started.**

Explanation: A start command for a service was sent to the service control manager (SCM).

System Action: The service process will start.

User Response: To check whether the service is running, reissue the service installation and administration program with the '-qss' argument.

CME2021E   **Cannot start service because the service executable file cannot be found.**

Explanation: The service executable was not found in the directories specified by the PATH environment variable.

System Action: The command was not processed.

User Response: Update the service with a fully-qualified service path, or add the service directory to the PATH environment variable.

CME2022E Cannot start service because service is marked for removal.

Explanation: The Windows service control manager (SCM) cannot start a service that has been marked for removal.

System Action: The command was not processed.

User Response: Install and then restart the service again.

CME2023E Cannot start service because service is already running.

Explanation: A service cannot be started twice.

System Action: The command was not processed.

User Response: None.

CME2024E Cannot start service due to invalid service dependencies.

Explanation: The service control manager (SCM) failed to start the dependent services.

System Action: The command was not processed.

User Response: Try to start all dependent services manually and reissue the command.

CME2025E Cannot start service because service has been disabled.

Explanation: The Windows service control manager (SCM) cannot start a service that has been disabled.

System Action: The command was not processed.

User Response: Enable the service with the Services applet and reissue the command.

CME2026E Cannot start service because service cannot be logged on.

Explanation: The Windows service control manager (SCM) cannot log on the service with the specified user account.

System Action: The command was not processed.

User Response: Check whether the specified user account has the right "Log on as service". If not, add this right to the account.

CME2027E Cannot start service due to a thread creation error.

Explanation: A new thread could not be created to run the service.

System Action: The command was not processed.

User Response: Reboot the system and reissue the command.

CME2028E Cannot start service due to a service initialization error.

Explanation: The service did not signal a successful start to the Windows service control manager (SCM) within the necessary time interval.

System Action: The command was not processed.

User Response: Check whether the service is installed with the correct parameters. If not, update the service installation. If so, reboot the system and reissue the command.

CME2029I Service has terminated.

Explanation: The Windows service control manager (SCM) sent a termination request to the service.

System Action: None.

User Response: To check whether the service has terminated, reissue the service installation and administration program with the '-qss' argument.

CME2030E Service cannot be stopped because other running services are dependent on it.

Explanation: The service cannot be stopped because other services that are running are dependent on it.

System Action: The command was not processed.

User Response: Terminate all dependent services, then reissue the command.

CME2031E Requested control code cannot be sent to the service.

Explanation: The requested control code cannot be sent to the service because the service has stopped or is pending.

System Action: The command was not processed.

User Response: Wait until the service has left its pending state. When service is running, reissue the command.

CME2032E Cannot stop service because service is not active.

Explanation: The service has not been started.

System Action: The command was not processed.

User Response: None.

CME2033E Cannot stop service because the system is shutting down.

Explanation: The Windows service control manager (SCM) cannot send a termination request to the service because the system is shutting down.

System Action: The command was not processed.

User Response: None.

CME2034E   Cannot stop service due to a general error.

Explanation: The Windows service control manager (SCM) could not send a termination request to the service.

System Action: The command was not processed.

User Response: To terminate the service, reboot the system.

CME2035I   Service was successfully updated.

Explanation: The service was successfully updated.

System Action: None.

User Response: None.

CME2036E   Cannot update service due to a circular dependency.

Explanation: The service control manager (SCM) cannot find a valid sequence to start all dependent services.

System Action: The command was not processed.

User Response: Check the specified dependency list and remove the circular dependency.

CME2037E   Cannot update service due to an invalid display name.

Explanation: The Windows service control manager (SCM) has rejected the specified service display name.

System Action: The command was not processed.

User Response: Reissue the command with a different service display name.

CME2038E   Cannot update service due to an invalid parameter.

Explanation: The service control manager (SCM) has rejected a specified service creation parameter.

System Action: The command was not processed.

User Response: Reissue the command with different parameters.

CME2039E   Cannot update service due to an invalid user account.

Explanation: The specified user account does not exist or has not the Windows system right ″Load and unload device drivers″ or ″Log on as a service″.

System Action: The command was not processed.

User Response: Reissue the command using a

different user account, or update the corresponding user rights.

CME2040E   Cannot update service because service is marked for removal.

Explanation: The service control manager (SCM) cannot update a service that is marked for removal.

System Action: The command was not processed.

User Response: Reinstall and then start the service.

CME2041W   Message catalog (DLL) not registered.

Explanation: No message catalog was registered to the Windows registry for service event logging.

System Action: None.

User Response: If a message catalog should be registered, update the service using the ′-sm′ Parameter.

CME2042E   User account *User_account* does not exist.

Explanation: The specified user account does not exist.

System Action: The command was not processed.

User Response: Reissue the command with a valid user account.

CME2043E   Due to an access denial, cannot check if user exists.

Explanation: Access is denied, cannot check if user exists.

System Action: The command was not processed.

User Response: Check user account and reboot the system.

CME2044E   Due to an invalid computer name, cannot check if user exists.

Explanation: The service installation program tried to check whether your user ID exists on your system.

System Action: The command was not processed.

User Response: Contact your IBM service representative.

CME2045E   Due to a general error, cannot check if user exists.

Explanation: Due to a general error, cannot check if user exists.

System Action: The command was not processed.

User Response: Reboot the system and reissue the command.

CME2046E   **Insufficient memory to process the function.**

Explanation: The memory is not sufficient to allocate necessary dynamic memory space.

System Action: The process will exit.

User Response: Stop other concurrent processes and reissue the command. If this does not work, increase operating system swap space.

CME2100E   **The specified profile has an invalid format.**

Explanation: The encryption utility has recognized that the content of the profile is not in the expected format.

System Action: The encryption utility terminates.

User Response: Check whether the name of the specified profile is correct. If so, refer to the user manual for more information about the profile format.

CME2101E   **Command line arguments are syntactically invalid.**

Explanation: The command line arguments are syntactically invalid.

System Action: The encryption utility terminates.

User Response: Call the program with parameter '-h' to get more information.

CME2102E   **Unable to open specified profile.**

Explanation: Unable to open specified profile. The profile was not found or was read-protected or write-protected.

System Action: The encryption utility terminates.

User Response: Check if path name and profile name are correct or remove read-protection or write-protection.

CME2103E   **No profile filename specified.**

Explanation: The user did not specify a profile file name.

System Action: The encryption utility will terminate.

User Response: Call the program with parameter '-h' to get more information.

CME2104E   **You entered different values.**

Explanation: The user entered two different keyword values.

System Action: The user is asked again to enter the keyword value (for example, the password) twice.

User Response: Re-enter the value twice.

CME2105E   **You did not enter a user ID.**

Explanation: The user did not enter a user ID.

System Action: The user is asked again to enter a valid user ID.

User Response: Enter a valid user ID.

CME2106E   **You did not enter a password.**

Explanation: The user did not enter a password.

System Action: The user is asked again to enter a valid password.

User Response: Enter a valid password.

CME2107E   **You did not enter a value.**

Explanation: User did not enter a value.

System Action: The user is asked again to enter a valid keyword value.

User Response: Enter a valid keyword value.

CME2108E   **You entered a user ID that is longer than *idlength* characters.**

Explanation: The user entered a user ID that is longer than 8 characters.

System Action: The user is asked to enter a new user ID.

User Response: Enter a new user ID that does not exceed 8 characters.

CME2109E   **You entered a password that is longer than *passwdlength* characters.**

Explanation: The user entered a password that is longer than 8 characters.

System Action: The user is asked to enter a new password.

User Response: Enter a new password that does not exceed 8 characters.

CME2110I   **Enter your user ID:**

Explanation: The user is requested to enter a user ID.

System Action: The system waits for user input.

User Response: Enter a user ID that does not exceed the maximum length.

CME2111I   Enter your password:

Explanation: The user is requested to enter a password.

System Action: The system waits for user input.

User Response: Enter a password that does not exceed the maximum length.

CME2112I   Keyword does not exist or contains no data. Enter the value to be encrypted:

Explanation: The user is requested to enter the value of the keyword.

System Action: The system waits for user input.

User Response: Enter the value of the keyword that is to be encrypted.

CME2113I   Re-enter your password:

Explanation: The user is requested to re-enter the password.

System Action: The system waits for user input.

User Response: Re-enter the password.

CME2114I   Re-enter your keyword value:

Explanation: The user is requested to re-enter the value of the keyword.

System Action: The system waits for user input.

User Response: Re-enter the value of the keyword.

CME2115I   Password successfully encrypted.

Explanation: The password was successfully encrypted, and the parameter CMEMIE_IM_PENC (for an import profile) or CMEMIE_EX_PENC (for an export profile) was added to the profile to indicate this.

System Action: None.

User Response: None.

CME2117W   Password already encrypted.

Explanation: The user tried to encrypt a password in a profile, but that password was already encrypted.

System Action: The encrypted password is left as it was.

User Response: None.

CME4000E   Invalid link number detected [link number *LinkNo*].

Explanation: An invalid link number was detected during MQIA processing.

System Action: The system received an invalid link number within a function call and terminates.

User Response: Internal error. Contact your local IBM representative.

CME4001E   Error creating message processing object [link number *LinkNo*].

Explanation: An internal error causes a failure when creating a message processing object for the specified link number. The reason is a memory allocation problem.

System Action: The system could not create a message processing object and terminates.

User Response: Verify system resources and try again.

CME4013W   Duplicate MERVA-related field in MQSeries message [MQSeries message ID x*MsgId*, MERVA-related field *Field*].

Explanation: A message received from the remote system contains duplicate MERVA-related fields. The message ID is shown in hexadecimal.

System Action: MQIA uses the contents of the last of the duplicate fields and ignores all previous.

User Response: Verify the remote application which created the message.

CME4100E   Unable to write log message to mailslot [OS Error *Code*].

Explanation: MQIA failed to write to the mailslot.

System Action: An internal system error prevents MQIA from writing to the mailslot. MQIA terminates.

User Response: Check the error code and resolve the problem.

CME4101E   Unable to open mailslot [OS Error *Code*].

Explanation: The log client could not open the mailslot to the log server.

System Action: An internal system error prevents MQIA from opening the mailslot for writing. MQIA terminates.

User Response: Check the error code and resolve the problem.

CME4102E   Unable to create event handle [OS Error *Code*].

Explanation: The event handle could not be created. Internal error.

System Action: An internal system error occurred while creating an event object. MQIA terminates.

User Response: Check the error code and resolve the problem.

CME4103E   Unable to open log file [filename *File*].

Explanation: The log server could not open the log file. The file name may be invalid or the file is already in use by another process or the file is write protected.

System Action: MQIA terminates.

User Response: Check file name and permissions.

CME4104E   Unable to create mailslot [OS Error *Code*].

Explanation: The MQIA log server was not able to create its mailslot.

System Action: MQIA terminates.

User Response: Check the error code and resolve the problem.

CME4105E   Unable to create activity event handle [OS Error *Code*].

Explanation: MQIA could not create an event handle.

System Action: MQIA terminates.

User Response: Check the error code and resolve the problem.

CME4106E   Unable to start log server thread.

Explanation: The log server thread did not start.

System Action: MQIA terminates.

User Response: Restart MQIA, if the problem persists, reboot your system.

CME4111E   Termination event handle invalid.

Explanation: The event handle for the program termination is invalid.

System Action: MQIA terminates.

User Response: Reboot your system. If the problem persists, contact your IBM representative.

CME4112E   Memory allocation failure.

Explanation: The required storage could not be allocated.

System Action: The system is running out of memory. MQIA terminates.

User Response: Verify your system resources and resolve the problem.

CME4114E   Log server termination event handle invalid [OS Error *Code*].

Explanation: The log server got an invalid event handle.

System Action: MQIA terminates.

User Response: Verify the OS error code and resolve the problem.

CME4117E   Invalid 'New-Log-Message' event handle [OS Error *Code*]. Log processing continues in polling mode.

Explanation: The event handle is not valid.

System Action: The log server continues its processing in polling mode. This means that the log message is written with a delay to the log file.

User Response: Verify the OS error code and resolve the problem.

CME4126E   Error writing to log file [OS Error *Code*].

Explanation: The log server could not write to the log file.

System Action: MQIA terminates.

User Response: Verify the OS error code and resolve the problem.

CME4136E   Unable to open Windows NT event log [OS Error *Code*].

Explanation: MQIA could not open the Windows NT event log.

System Action: MQIA terminates.

User Response: Verify the OS error code and resolve the problem.

CME4139E   Log Server did not terminate within specified time frame of *Timeout* sec, thread will be killed now.

Explanation: During MQIA shutdown the log server did not terminate properly.

System Action: The system kills the log server after the timeout period.

User Response: Verify the Windows NT event log entries for possible reasons.

CME4149E   Log Server terminated abnormally.

Explanation: The log server thread terminated due to a severe error.

System Action: MQIA terminates.

User Response: Check the previous error log

messages to determine why the log server terminated.

CME4200E   MQ2MV process could not connect to MV2MQ process [OS Error *Code*].

Explanation: The MQ2MV process tries to check if the MV2MQ process is running. This check fails.

System Action: MQIA terminates.

User Response: Check the OS error code, resolve the problem, and restart MQIA.

CME4202E   Error loading MQSeries *LibraryType* library [OS Error *Code*].

Explanation: MQIA could not load the MQSeries client or server DLL.

System Action: MQIA terminates.

User Response: Check your environment settings and the availability of the MQSeries DLLs.

CME4206W   MQ2MV process did not terminate within specified time frame. MQ2MV process is killed now [timeout *Timeout* sec].

Explanation: The MQ2MV process of MQIA did not terminate within the specified timeout interval.

System Action: MQIA kills the hanging process.

User Response: Verify MQIA and/or Windows NT event log entries for possible reasons.

CME4211E   Error loading MERVA API library [OS Error *Code*].

Explanation: The MERVA API DLL could not be loaded by MQIA.

System Action: MQIA terminates.

User Response: Check your environment settings and the availability of the MERVA API DLL.

CME4214E   Configuration file not accessable, terminating MQIA [filename *MQIAProfile*].

Explanation: MQIA could not read the contents of the specified configuration file.

System Action: MQIA terminates.

User Response: Check if the specified configuration file is available and syntactically correct.

CME4215E   Could not create event handle for termination response [OS Error *Code*].

Explanation: MQIA could not create an event handle.

System Action: MQIA terminates.

User Response: Verify the OS error code and resolve the problem. Restart MQIA. If the problem persists, contact your local IBM representative.

CME4217E   Could not create MQ2MV process [OS Error *Code*, command string *Command*].

Explanation: MQIA tried to create the MQ2MV process by applying the denoted command string.

System Action: MQIA terminates.

User Response: Verify the OS error code and the command string. If possible, resolve the problem, otherwise contact your local IBM representative.

CME4220E   Log Server not running (Activity check failed).

Explanation: The log server is not running.

System Action: MQIA terminates.

User Response: Verify MQIA and Windows NT event logs for possible reasons. If the problem persists, contact your local IBM representative.

CME4236E   Unable to initialize parent object.

Explanation: The parent object could not be initialized properly.

System Action: Internal error. MQIA could not initialize its parent object and terminates.

User Response: Verify MQIA and Windows NT event logs for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4240E   Unable to create log server object [OS Error *Code*].

Explanation: The log server object could not be created.

System Action: MQIA could not create the log server object and terminates.

User Response: Verify the OS error code and resolve the problem. If the problem persists, contact your local IBM representative.

CME4241E   Unable to install console control handler [OS Error *Code*].

Explanation: The console control handler could not be installed.

System Action: MQIA terminates.

User Response: Verify OS error code and resolve the problem. If the problem persists, contact your local IBM representative.

CME4242E   Error creating termination event handle [OS Error *Code***].**

Explanation: The termination event handle could not be created.

System Action: MQIA terminates.

User Response: Verify the OS error code and resolve the problem. If the problem persists, contact your local IBM representative.

CME4251W   Coordinator thread did not terminate within specified time frame of *Timeout* **sec. Coordinator thread is killed now.**

Explanation: The coordinator thread did not terminate within the specified timeout value.

System Action: MQIA kills the hanging thread and terminates.

User Response: Verify MQIA and Windows NT event log for possible reasons.

CME4254E   Unable to create coordinator object.

Explanation: The coordinator object could not be created.

System Action: MQIA terminates.

User Response: Verify MQIA and Windows NT event log for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4257E   Unable to create coordinator thread.

Explanation: The coordinator thread could not be started.

System Action: MQIA terminates.

User Response: Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4258E   Coordinator thread did not start within specified time frame of *Timeout* **sec.**

Explanation: The coordinator thread did not start within the specified timeout period.

System Action: MQIA terminates.

User Response: Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4260E   Internally used service program name corrupted [Service name *ServiceName***].**

Explanation: The program name of the service could not be retrieved.

System Action: MQIA terminates.

User Response: Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4261E   MQ2MV process not running! MQIA will be terminated now.

Explanation: MQIA has detected that the MQ2MV process is not running.

System Action: MQIA terminates.

User Response: Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4306E   Transaction completed with error.

Explanation: An MQIA transaction completed with an error.

System Action: MQIA terminates.

User Response: Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4309E   Transaction object could not be created, NULL pointer received.

Explanation: The transaction object could not be created. The creation returned a NULL pointer.

System Action: MQIA terminates.

User Response: Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4326E   Error in MERVA communication. MQIA terminates [MERVA error *Code***].**

Explanation: A MERVA API call failed with a MERVA error.

System Action: MQIA terminates.

User Response: Verify MERVA error code and resolve the problem.

CME4342E　MQIA MV2MQ process no longer available.

Explanation:　The MV2MQ process has terminated.

System Action:　MQIA terminates.

User Response:　Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4344E　Phase II Recovery failed, terminate MQIA.

Explanation:　Phase II of the recovery processing failed.

System Action:　MQIA terminates.

User Response:　Verify MQIA log and Windows NT event log entries for possible reasons and try again. If the problem persists, contact your local IBM representative.

CME4346E　Recovery Phase I completed with error, terminate MQIA.

Explanation:　Phase I of the recovery processing failed.

System Action:　MQIA terminates.

User Response:　Verify MQIA log and Windows NT event log entries for possible errors and try again. This message also appears if the user terminates MQIA and MQIA recovery processing is still active. If the recovery processing does not end in a specific amount of time, check the routing conditions of all MERVA receive and wait queues. Note that all complete messages must be routed to a destination queue other than the origin MERVA queue.

CME4401E　Memory reallocation error.

Explanation:　The required memory could not be allocated.

System Action:　The system is running out of memory. MQIA terminates.

User Response:　Verify your system resources and resolve the problem.

CME4407E　Invalid process type specified [Process type = *ProcType*].

Explanation:　Internal error. The process type is not known to MQIA.

System Action:　MQIA terminates.

User Response:　Verify MQIA log and Windows NT event log entries for possible reasons and contact your local IBM representative.

CME4501E　Duplicate link section headings encountered [filename *Profile*, link number *LinkNo*, name *LinkName*].

Explanation:　Two or more link-specific sections with the same heading found in the configuration file.

System Action:　MQIA terminates.

User Response:　Make sure that the link-specific section headings are unique within the configuration file.

CME4505E　No log directory specified in configuration file [filename *Profile*, key *LogDir*].

Explanation:　No directory for the MQIA log file specified in the configuration file.

System Action:　MQIA terminates.

User Response:　Specify a valid directory for logging in the configuration file.

CME4506E　MERVA receive queue definition contains an invalid queue type [filename *Profile*, link *Link*, queue *Queue*, type *Type*].

Explanation:　An invalid MERVA receive queue definition encounterd.

System Action:　MQIA terminates.

User Response:　Correct the MERVA receive queue parameter in the configuration file.

CME4507E　Unable to create log client.

Explanation:　The log client could not be created.

System Action:　MQIA could not create the log client object and terminates.

User Response:　Check the event log and the MQIA log file for possible reasons. If the problem persists, contact your local IBM representative.

CME4508E　Length of MERVA send queue name is invalid. [filename *Profile*, link *Link*, send queue no *QueueNo*, queue name *Name*, valid length 8].

Explanation:　The length of the name for the MERVA send queue is not valid.

System Action:　MQIA terminates.

User Response:　Specify a valid MERVA send queue name according to the MERVA naming rules.

CME4509E **Length of the semaphore name for MERVA send queue is not valid. [filename** *Profile*, **link** *Link*, **send queue no** *QueueNo*, **semaphore name** *Name*, **valid length 8].**

Explanation: The length of the semaphore name in the MERVA send queue definition is not valid.

System Action: MQIA terminates.

User Response: Specify a valid MERVA semaphore name according to the MERVA naming rules which is assigned to the appropriate MERVA send queue.

CME4510E **The message priority in the MERVA send queue definition is invalid [filename** *Profile*, **link** *Link*, **send queue no** *QueueNo*, **value** *Value*].

Explanation: The message priority value for the MERVA send queue is not valid.

System Action: MQIA terminates.

User Response: Correct the message priority in the MERVA send queue definition. It must be in the range of 0 to 9.

CME4511E **Report option in the MERVA send queue definition is invalid. [filename** *Profile*, **link** *Link*, **send queue no** *QueueNo*, **value** *Value*].

Explanation: The specified value for the report option in the MERVA send queue definition is not valid.

System Action: MQIA terminates.

User Response: Correct the report option. It must be an integer value between 0 and 15.

CME4512E **Field option in MERVA send queue definition is invalid [filename** *Profile*, **link** *Link*, **send queue no** *QueueNo*, **value** *Value*].

Explanation: The specified value for the field option in the MERVA send queue definition is not valid.

System Action: MQIA terminates.

User Response: Correct the field option. It must be an integer value between 0 and 511.

CME4513E **Missing MERVA send queue definition [filename** *Profile*, **link** *Link*, **send queue no** *QueueNo*].

Explanation: The name of the MERVA send queue is not specified.

System Action: MQIA terminates.

User Response: Specify a valid name for the MERVA

send queue in the configuration file.

CME4514E **Missing semaphore name in MERVA send queue definition [filename** *Profile*, **link** *Link*, **queue** *QueueNo*].

Explanation: No semaphore name is assigned to the MERVA send queue in the configuration file.

System Action: MQIA terminates.

User Response: Add a valid semaphore name to the MERVA send queue definition in the configuration file.

CME4515E **Missing message priority in MERVA send queue definition [filename** *Profile*, **link** *Link*, **queue** *QueueNo*].

Explanation: No message priority is assigned to the MERVA send queue in the configuration file.

System Action: MQIA terminates.

User Response: Add a valid message priority to the MERVA send queue definition in the configuration file.

CME4516E **Missing report option in MERVA send queue definition [filename** *Profile*, **link** *Link*, **queue** *QueueNo*].

Explanation: No report option is assigned to the MERVA send queue in the configuration file.

System Action: MQIA terminates.

User Response: Add a valid report option to the MERVA send queue definition in the configuration file.

CME4517E **Scan interval is out of range [filename** *Profile*, **specified value** *SpecifiedValue* **sec].**

Explanation: The specified value for the scan interval is out of range.

System Action: MQIA terminates.

User Response: Correct the scan interval. The value must be in the range of 10 to 3600 seconds.

CME4518E **Invalid log level specified [filename** *Profile*, **specified log level** *SpecifiedLevel*].

Explanation: The specified value for the log level is out of range.

System Action: MQIA terminates.

User Response: Specify a value within the valid range (1 to 4) in the configuration file.

CME4520A   No MERVA user name and password specified, user logon required.

Explanation: Neither a MERVA user name nor a password were specified at the command line, or in the configuration file. The user is requested to enter a user name and password.

System Action: The user is asked to enter a MERVA user name and password.

User Response: Enter a valid MERVA user name and password.

CME4521A   No MERVA password specified, user logon required.

Explanation: No MERVA password was specified at the command line or in the configuration file.

System Action: The user is asked to enter a valid password for the given MERVA user name.

User Response: Enter the valid MERVA password.

CME4524W   MERVA application name in configuration file not specified [filename *Profile***, default MERVA application name** *ApplicationID***].**

Explanation: No MERVA application name was specified in the configuration file.

System Action: MQIA uses the default value.

User Response: If the default value is not desired, you must specify a valid MERVA application name in the configuration file.

CME4526W   Default log level used [filename *Profile***, LogLevel** *LogLevel***].**

Explanation: The specified log level is not valid or no log level was found in the configuration file.

System Action: MQIA uses log level 1.

User Response: If the default value is not desired, add a valid MERVA log level (1 to 4) to the configuration file.

CME4527W   No MERVA APITrace parameter was found in the configuration file [filename *Profile***, default API Trace** *APITrace***].**

Explanation: No MERVA APITrace parameter was found in the configuration file.

System Action: MQIA uses the default 'No'.

User Response: If the default value is not desired, add a valid MERVA APITrace option (YES or NO) to the configuration file.

CME4528W   MQ2MV polling not specified [filename *Profile***, default value** *Value***].**

Explanation: No MQ2MV polling parameter was found in the configuration file.

System Action: MQIA uses the default.

User Response: If the default value is not desired, add a valid MQ2MV polling parameter (YES or NO) to the MQIA configuration file.

CME4529W   MV2MQ polling not specified [filename *Profile***, default value** *Value***].**

Explanation: No MV2MQ polling parameter was found in the configuration file.

System Action: MQIA uses the default.

User Response: If the default value is not desired, add a valid MV2MQ polling parameter (YES or NO) to the MQIA configuration file.

CME4530W   Scan interval parameter not specified or invalid [filename *Profile***, default scan interval** *ScanInterval* **sec].**

Explanation: The scan interval parameter in the configuration file is not specified or is invalid.

System Action: MQIA uses the default value.

User Response: If the default value is not desired, add a valid value for the scan interval (10 - 3600 sec) to the configuration file.

CME4531E   MQSeries Reference queue not specified [filename *Profile***, key** *Key***].**

Explanation: The parameter for the MQSeries reference queue is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MQSeries queue as MQSeries reference queue to the configuration file.

CME4532E   MQSeries initiation queue not specified [filename *Profile***, key** *Key***].**

Explanation: The parameter for the MQSeries initiation queue is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MQSeries queue as MQSeries initiation queue to the configuration file.

CME4533E   MQSeries queue manager not specified [filename *Profile***, key** *Key***].**

Explanation: The parameter for the MQSeries queue manager is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MQSeries queue manager name to the configuration file.

CME4534W   MQSeries reference queue is also used as TransAct queue [filename *Profile*, queue name *QueueName*].

Explanation: The parameter for the MQSeries transaction queue is not specified or is empty.

System Action: MQIA continues processing and uses the MQSeries reference queue as MQSeries transaction queue too.

User Response: If a separate MQSeries transaction queue is desired, add a valid MQSeries queue name as MQSeries TransAct queue to the configuration file.

CME4535W   Common remote user name is used in link [filename *Profile*, link *Link*, RmtUser *Username*].

Explanation: No remote user name specified for the denoted link.

System Action: MQIA uses the remote user name specified in the common section as remote user name for the current link.

User Response: If a link-specific remote user name is desired, add a valid remote user name to the denoted link section in the configuration file.

CME4536E   MQSeries receive queue not specified [filename *Profile*, link *Link*, key *Key*].

Explanation: The parameter for the MQSeries receive queue of the denoted link is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MQSeries queue as MQSeries receive queue to the denoted link section in the configuration file.

CME4537E   MQSeries send queue not specified [filename *Profile*, link *Link*, key *Key*].

Explanation: The parameter for the MQSeries send queue of the denoted link is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MQSeries queue as MQSeries send queue to the denoted link section in the configuration file.

CME4538W   Parameter of the link-specific MQ2MV polling is not specified or invalid, the value of the common section is used [filename *Profile*, link *Link*, value *Value*].

Explanation: The link-specific MQ2MV polling option is invalid or not specified.

System Action: MQIA sets the link-specific MQ2MV polling mode to the value of the MQ2MV polling mode specified in the common section.

User Response: If the value of the common section is not desired, add a valid MQ2MV polling parameter (YES or NO) to the appropriate link section in the configuration file.

CME4539W   Parameter of the link-specific MV2MQ polling is not specified or invalid, the value of the common section is used [filename *Profile*, link *Link*, value *Value*].

Explanation: The link-specific MV2MQ polling option is invalid or not specified.

System Action: MQIA sets the link specific MV2MQ polling mode to the value of the MV2MQ polling mode specified in the common section.

User Response: If the value of the common section is not desired, add a valid MV2MQ polling parameter (YES or NO) to the appropriate link section in the configuration file.

CME4540E   MERVA receive queue not specified [filename *Profile*, link *Link*, key *Key*].

Explanation: The parameter for the MERVA receive queue of the denoted link is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MERVA queue for the MERVA receive queue in the configuration file.

CME4541E   MERVA wait queue not specified [filename *Profile*, link *Link*, key *Key*].

Explanation: The parameter for the MERVA wait queue of the denoted link is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid MERVA queue for the MERVA wait queue in the configuration file.

CME4542E   No MERVA send queue specified [filename *Profile*, link *Link*, key *Key*].

Explanation: The parameter for the MERVA send queue of the denoted link is not specified or is empty.

System Action: MQIA terminates.

User Response: Add a valid definition for the MERVA send queue in the configuration file.

CME4543E   Not all MERVA send queues defined correctly [filename *Profile*, link *Link*, key *Key*].

Explanation: One or more MERVA send queues in the denoted link are not defined properly.

System Action: MQIA terminates.

User Response: Make sure that all MERVA send queues are defined properly.

CME4544E Configuration file not found [filename *Profile***].**

Explanation: MQIA did not find the specified configuration file.

System Action: MQIA terminates.

User Response: Specify a valid configuration file name, or make sure that a configuration file with the same name as the currently active MERVA instance is located in the MERVA instance directory.

CME4545E Common section not found in configuration file [filename *Profile***, key** *SectionName***].**

Explanation: No common section definition found in configuration file.

System Action: MQIA terminates.

User Response: Define a valid common section in the MQIA configuration file with all mandatory parameters.

CME4546E Invalid value for key specified, only Y(es) or N(o) allowed [filename *Profile***,** *Link Name***, key** *Key***, specified value** *Value***].**

Explanation: The specified value is not allowed for the denoted parameter.

System Action: MQIA terminates.

User Response: Correct the parameter. Valid values are Y(es) or N(o).

CME4547E Invalid SWIFT message check option for MERVA *QueueType* **queue specified, only Y(es) or N(o) allowed [filename** *Profile***, link** *LinkName***, key** *Key***, specified value** *Value***].**

Explanation: The option for SWIFT message checking is invalid.

System Action: MQIA terminates.

User Response: Only Y(es) or (N)o are allowed for the SWIFT message check option, correct the settings in the configuration file.

CME4548W Invalid key in common section of configuration file [filename *Profile***, key** *Key***].**

Explanation: An unknown parameter was defined in the common section.

System Action: MQIA continues processing.

User Response: Remove the unknown parameter from the common section in the configuration file or verify if it is a typing error and correct the mistake.

CME4549E Missing value for key in common section of configuration file [filename *Profile***, key** *Key***].**

Explanation: A mandatory parameter is missing in the common section.

System Action: MQIA terminates.

User Response: Add a value to the key in the common section.

CME4550E No data in link specific section in configuration file found [filename *Profile***, link** *Link***].**

Explanation: The specified link specific section contains no data.

System Action: MQIA terminates.

User Response: Define all mandatory parameters in the link specific section.

CME4551E Memory allocation error during configuration data loading [filename *Profile***].**

Explanation: The required memory could not be allocated for loading the configuration data.

System Action: The system is running out of memory. MQIA terminates.

User Response: Verify your system resources and resolve the problem. Check the used configuration file.

CME4552W No value specified for parameter [filename *Profile***,** *Section Type***, key** *Key***].**

Explanation: No value is specified for parameter in the denoted section.

System Action: MQIA continues processing.

User Response: Add a valid value for the denoted parameter.

CME4553W Invalid key in link specific section of configuration file [filename *Profile***, link** *LinkName***, key** *Key***].**

Explanation: A unknown parameter was found in a link specific section.

System Action: MQIA continues processing.

User Response: Remove the unknown parameter from the link specific section or verify if it is a typing error and correct the mistake.

CME4555E   Definition in wrong section [filename *Profile*, *Section Type*, **key** *Key***].**

Explanation: Internal error. A key is used in an invalid section.

System Action: MQIA terminates.

User Response: Verify the MQIA and Windows NT event log for a possible reason. Contact your local IBM representative.

CME4556E   No value specified for key [filename *Profile*, *Section Type*, **key** *Key***].**

Explanation: No value is specified for the denoted key in the specified section.

System Action: MQIA terminates.

User Response: Add a valid value for the denoted key.

CME4557E   Invalid name length [filename *Profile*, *Section Type*, **key** *Key*, **valid length** *Length*, **name** *Name***].**

Explanation: The specified name value of the denoted key is longer than the allowed maximum length.

System Action: MQIA terminates.

User Response: Specify the value within the maximum length.

CME4558W   Value of parameter in common section redefined [filename *Profile*, **key** *Key*, **old value** *OldValue*, **new value** *NewValue***].**

Explanation: A previously defined value of the denoted key in the common section is redefined.

System Action: MQIA continues processing and uses the new value.

User Response: To avoid this warning message, remove all duplicate definitions within the common section.

CME4559W   Value of parameter in link section redefined [filename *Profile*, **link** *Link*, **key** *Key*, **old value** *OldValue*, **new value** *NewValue***].**

Explanation: A previously defined value of the denoted key in the specified link is redefined.

System Action: MQIA continues processing and uses the new value.

User Response: To avoid this warning message, remove all duplicate definitions within the link section.

CME4560E   Invalid data key specified [filename *Profile*, *Section Type*, **key** *Key***].**

Explanation: Internal error. An invalid internal data key is used.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4561E   Key in common section is a string type, not a number [filename *Profile*, **key** *Key***].**

Explanation: Internal error. The system tried to use a key of type string in the common section as a number.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4562E   Key in denoted section is a string type, not a number [filename *Profile*, *Section Type*, **key** *Key***].**

Explanation: Internal error. The system tried to use a key of type string in the denoted section as a number.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4564E   Buffer too small to get data [key *Key*, *Section Type*, **required buffer size** *RequiredSize*, **available buffer size** *AvailableSize***].**

Explanation: Internal error. The provided buffer is too small for the denoted key value to get.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4568E   Buffer too small for data [key *Key*, **link** *Link*, **required buffer size** *RequiredSize*, **available buffer size** *AvailableSize***].**

Explanation: Internal error. The provided buffer is too small for the value of the denoted key to get.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4570E   Invalid data key detected [filename *Profile*, *Section Type*, **key** *Key***].**

Explanation: An invalid key was requested in denoted section.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4572E **Invalid link number detected [link number** *LinkNo***, max. valid link number** *MaxLinkNo***].**

Explanation: Internal error. An invalid link number was encountered.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4573E **MERVA send queue number out of range [specified** *MERVASendQNo***, max. valid** *MaxValidNo***].**

Explanation: Internal error. An invalid MERVA send queue number was encountered.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4576E **Error writing key to configuration file [filename** *Profile***, key** *Key***, OS error** *Code***].**

Explanation: The specified parameter *Key* and its value could not be written to the configuration file.

System Action: MQIA continues processing but this error may cause a termination later on.

User Response: Verify the OS error code and resolve the problem.

CME4578E **Key not found in common section [filename** *Profile***, key** *Key***].**

Explanation: The specified parameter *Key* is not found in the common section of the configuration file.

System Action: MQIA continues processing.

User Response: Add the parameter *Key* to your configuration file.

CME4579E **Error loading configuration data.**

Explanation: The configuration data could not been loaded.

System Action: MQIA is unable to continue processing and terminates.

User Response: Verify the MQIA log and Windows NT event log for possible reasons.

CME4581W **No conversion exit name specified in configuration file** *Profile* **[link** *Link***, format** *UsedFormat***].**

Explanation: No conversion exit name is specified for the denoted link. Thus, no conversion is performed in any MQSeries conversion exit.

System Action: The MQSeries default non-string

format is inserted as message format in the MQMD.

User Response: If conversion in any MQSeries conversion exit on any remote system is required, the appropriate conversion exit name must be specified with this key.

CME4582W **Field options not specified for MQIA MERVA send queue [filename** *Profile***, link** *Link***, queue** *MERVASendQ***, field options** *FieldOptions***].**

Explanation: No field options specified for denoted MERVA send queue.

System Action: MQIA uses the default value, which means, 'all fields'.

User Response: If only a specific set of fields is required, define the field options in the appropriate way.

CME4583W **Exception report as the only report option is not valid for a MERVA send queue, add COA [filename** *Profile***, link** *Link***, queue** *MERVASendQ***, report options COA-EXCEPTION].**

Explanation: Invalid report option specified. Exception report as the only report option is not valid.

System Action: MQIA uses exception report in conjunction with COA report option.

User Response: If report options other than COA-EXCEPTION are required, change the configuration file.

CME4584E **Duplicate MQSeries queue name [filename** *Profile***,** *Section Type***, key** *Key***, duplicate in** *Section Type***, key** *Key***, name** *QueueName***].**

Explanation: The specified MQSeries queue name is used more than once in the configuration file.

System Action: MQIA terminates.

User Response: Define unique MQSeries queues in the configuration file.

CME4585E **Duplicate MERVA queue name [filename** *Profile***, link** *Link***, key** *Key***, duplicate in link** *Link***, key** *Key***, name** *QueueName***].**

Explanation: The specified MERVA queue name is used more than once in the configuration file.

System Action: MQIA terminates.

User Response: Define unique MERVA queues in the configuration file.

CME4587W   **Old MERVA receive queue definition detected, use receive queue as datagram receive queue [filename** *Profile*, **link** *Link*, **queue** *QueueName*].

Explanation: A former MERVA receive queue definition was detected.

System Action: The queue is used as datagram receive queue.

User Response: Change the MERVA receive queue definition in the configuration file to the new notation.

CME4588W   **No MERVA datagram receive queue specified [filename** *Profile*, **link** *Link*, **key** *Key*, **queue** *QueueName*].

Explanation: No MERVA receive queue with queue type datagram defined in the denoted link.

System Action: MQIA uses the MERVA receive queue as a request and datagram receive queue.

User Response: If you want to differentiate in MERVA between received datagrams and requests, define a MERVA receive queue with queue type datagram and a MERVA receive queue with queue type request in the configuration file.

CME4589W   **No MERVA request receive queue specified [filename** *Profile*, **link** *Link*, **key** *Key*, **queue** *Queue*].

Explanation: No MERVA receive queue with queue type request defined in the denoted link.

System Action: MQIA uses the MERVA receive queue as request and datagram receive queue.

User Response: If you want to differentiate in MERVA between received datagrams and request, you must define a MERVA receive queue with queue type datagram as well as a MERVA receive queue with queue type request in the configuration file.

CME4590W   **MERVA datagram and request receive queue name are indentical [filename** *Profile*, **link** *Link*, **key** *Key*, **duplicate in link** *Link*, **key** *Key*, **name** *QueueName*].

Explanation: The MERVA receive queue is used for both, request and datagram messages.

System Action: MQIA continues processing.

User Response: None.

CME4591W   **MQSeries reference queue is also used as MQSeries transaction queue [filename** *Profile*, **common section, key** *Key*, **also used in common section, key** *Key*, **name** *QueueName*].

Explanation: There is no MQSeries transaction queue defined.

System Action: MQIA uses the MQSeries reference queue as MQSeries transaction queue too.

User Response: If a separate MQSeries transaction queue is required, define it in the common section of the configuration file.

CME4592E   **No link-specific section found in configuration file [filename** *Profile*].

Explanation: No link-specific section defined in the configuration file.

System Action: MQIA terminates.

User Response: At least one link-specific section must be defined in the configuration file.

CME4593E   **More than one common section definition found in configuration file [filename** *Profile*, **key** *SectionName*].

Explanation: The configuration file contains more than one common section definition.

System Action: MQIA terminates.

User Response: Only one common section is allowed in the configuration file.

CME4594W   **Duplicate link section name [filename** *Profile*, **link number** *Link*, **key** *Key*, **also used in link number** *Link*, **name** *QueueName*].

Explanation: Duplicate link name found in configuration file.

System Action: MQIA continues processing.

User Response: To easily identify the link in case of an error it is a good practice to have unique link names defined in the configuration file.

CME4595E   **Value of MQIA key in common section redefined [filename** *Profile*, **key** *Key*, **old value** *OldValue*, **new value** *NewValue*].

Explanation: A previously defined value of the denoted key in the common section is redefined.

System Action: MQIA terminates.

User Response: Remove all duplicate definitions within the common section.

CME4596E   **Value of MQIA key in link section redefined [filename** *Profile*, **link** *Link*, **key** *Key*, **old value** *OldValue*, **new value** *NewValue*].

Explanation: A previously defined value of the denoted key in the specified link is redefined.

System Action: MQIA terminates.

User Response: Remove all duplicate definitions within the link section.

CME4610A   **Terminate MQIA by pressing CRTL-C**

Explanation: To end MQIA press CTRL-C

System Action: A popup window is displayed.

User Response: Select Yes to end MQIA or No to continue processing.

CME4700E   **Error opening MQSeries queue [queue manager** *QueueMgr*, **queue name** *QueueName*, *Section Type*, **completion** *Code*, **reason** *Code*].

Explanation: An attempt to open denoted MQSeries queue failed.

System Action: MQIA terminates.

User Response: Verify the queue name and that the queue manager is running. If it is a client connection, verify that the MQSeries client channel and the MQSeries client environment variables are defined properly.

CME4702E   **Error creating MQIA resource object.**

Explanation: The MQIA resource object could not be created.

System Action: MQIA terminates.

User Response: Verify MQIA and Windows NT event log files for possible reasons and try again. If the problem persists, reboot your system and try again.

CME4706E   **MQSeries resource verification failed.**

Explanation: Not all MQSeries resources are defined well or are available.

System Action: MQIA terminates.

User Response: Check the used MQSeries objects and the definitions in the configuration file. Verify MQIA log file for possible reasons.

CME4709E   **Opening MQSeries resources failed.**

Explanation: One or more MQSeries resources could not be opened.

System Action: MQIA terminates.

User Response: Check the used MQSeries objects and the definitions in the configuration file. Verify MQIA log file for possible reasons.

CME4711E   **Error setting MERVA application name [MERVA error** *Code*].

Explanation: An attempt to set the MERVA application name failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4712E   **Error attaching to MERVA [MERVA error** *Code*].

Explanation: An attempt to attach to MERVA failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4715E   **Currently not attached to MERVA, therefore unable to create MERVA send queues semaphore list.**

Explanation: MQIA is not attached to MERVA and is therefore not able to create a MERVA send queue semaphore list.

System Action: MQIA terminates.

User Response: Verify the MQIA log file for possible reasons and try again.

CME4717E   **Error opening semaphore of MERVA send queue [link** *Link*, **send queue** *QueueName*, **semaphore** *Semaphore*, **MERVA error** *Code*].

Explanation: The denoted sempahore could not be opened.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4718E   **Error adding MERVA semaphore to semaphore list [link** *Link*, **send queue** *QueueName*, **semaphore** *Semaphore*, **OS Error** *Code*].

Explanation: The denoted semaphore could not be added to the list.

System Action: MQIA terminates.

User Response: Verify the OS error code and the MQIA log file. If the error persists, contact your local IBM representative.

CME4721E   Error adding MQIA termination event handle to MERVA semahore list.

Explanation: Internal error. The termination event handle could not be added to the semaphore list.

System Action: MQIA terminates.

User Response: Verify the MQIA log file and restart MQIA. If the error persists, contact your local IBM representative.

CME4724E   Invalid link number or resource type specified [link number *Link*, **resource type** *Resource*].

Explanation: The link number or the resource type encountered is invalid.

System Action: MQIA terminates.

User Response: Verify the MQIA log file for a possible reason.

CME4725E   Invalid MERVA queue specified [link *Link*, **queue name** *QueueName*].

Explanation: The MERVA queue specified is not valid for this link.

System Action: MQIA terminates.

User Response: Check definitions of the denoted link.

CME4726E   Invalid link number or MERVA send queue number specified [link number *Link*, **MERVA send queue no** *QueueName*].

Explanation: The link number or the MERVA send queue number is invalid.

System Action: MQIA terminates.

User Response: Verify the MQIA log file for possible reasons.

CME4727E   The specified queue is no MERVA send queue [link *Link*, **queue name** *QueueName*].

Explanation: The denoted queue is not defined in MERVA.

System Action: MQIA terminates.

User Response: Correct the MERVA send queue definitions of the denoted link in the configuration file or define the queue in MERVA.

CME4728E   Invalid MERVA send queue routing detected [link *Link*, **queue name** *QueueName*].

Explanation: The following routing condition for the MERVA send queue is missing: Route to MERVA wait queue if MSGOK = 'MVQWAIT'.

System Action: MQIA terminates.

User Response: Check the MERVA send queue routing of the denoted queue in the MERVA customization. For more information see *MQI Attachment User's Guide*.

CME4731E   Error connecting to MQSeries queue manager *ConnectionType* **[queue manager** *QueueManager*, **completion** *Code*, **reason** *Code*].

Explanation: An attempt to connect to the denoted MQSeries queue manager failed.

System Action: MQIA terminates.

User Response: Check your queue manager name in the configuration file, verify that the queue manager is running and, if it is an MQSeries client connection, make sure that the MQSeries client channel and the environment variables are set properly.

CME4736E   Error opening MQSeries queue manager for retrieving dead-letter queue name [queue manager *QueueMgr*, **completion** *Code*, **reason** *Code*].

Explanation: The MQSeries queue manager could not be opened to retrieve the dead-letter queue name.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4737E   Error retrieving MQSeries dead-letter queue name [queue manager *QueueMgr*, **completion** *Code*, **reason** *Code*].

Explanation: The MQSeries dead-letter queue name could not be retrieved.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4746E   Message could not be retrieved successfully from MQSeries queue [_Section Type_, **queue name** *QueueName*, **completion** *Code*, **reason** *Code*].

Explanation: An attempt to get a message from the denoted MQSeries queue failed.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4748E   Message could not be put on MERVA queue [*Section Type*, queue name *QueueName*, MRN *MRN*, MERVA error *Code*].

Explanation: An attempt to put the message with the message reference number MRN on the denoted MERVA queue failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4751E   Message could not be retrieved from MERVA queue [*Section Type*, queue name *QueueName*, MERVA error *Code*].

Explanation: An attempt to get a message from the denoted MERVA queue failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4755E   Message could not be retrieved from MERVA send queue [link *Link*, queue name *QueueName*, MERVA error *Code*].

Explanation: An attempt to get a message from the denoted MERVA queue failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4759E   Message could not be put on MERVA send queue [link *Link*, queue name *QueueName*, MRN *MRN*, MERVA error *Code*].

Explanation: An attempt to put the message with the message reference number MRN on the denoted MERVA send queue failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4760E   Message could not be put on MQSeries queue [*Section Type*, queue name *QueueName*, MsgId x*MsgId*, completion *Code*, reason *Code*].

Explanation: An attempt to put the message with the message identifier xMsgId (hexadecimal) on the denoted MQSeries queue failed.

System Action: MQIA terminates.

CME4765E   MQSeries COMMIT operation failed [completion *Code*, reason *Code*].

Explanation: The MQSeries commit operation failed.

System Action: MQIA performs an MQSeries backout, this means all messages remain in their source queues. MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4768E   MQIA transaction could not be committed [MERVA error *Code*].

Explanation: The message could not be routed to its MERVA destination queue or the message could not be unlocked.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4771E   MQSeries BACKOUT operation failed [completion *Code*, reason *Code*].

Explanation: The MQSeries backout operation failed.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4773E   MQIA transaction could not be rolled back [MERVA error *Code*].

Explanation: The message could not be unlocked in the MERVA queue.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4776E   Error putting message on MQSeries dead-letter queue [queue manager *QueueMgr*, queue name *QueueName*, completion *Code*, reason *Code*].

Explanation: An attempt to put a message on the MQSeries dead-letter queue failed.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4779E Error opening MQSeries queue [*Section Type*, queue name *QueueName*, completion *Code*, reason *Code*].

Explanation: An attempt to open the denoted MQSeries queue failed.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4780E Error retrieving queue depth of MQSeries queue [*Section Type*, queue name *QueueName*, completion *Code*, reason *Code*].

Explanation: An attempt to query the denoted MQSeries queue for the queue depth failed.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4783E Queue depth on MERVA queue could not be determined [*Link Name*, queue name *QueueName*, MERVA error *Code*].

Explanation: The queue depth of the denoted MERVA queue could not be retrieved.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4786E Queue depth on MERVA send queue could not be determined [link *Link*, queue name *QueueName*, MERVA error *Code*].

Explanation: The queue depth of the denoted MERVA send queue could not be retrieved.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4788E Invalid link number or MERVA queue type specified [link number *LinkNumber*, queue type *QueueType*].

Explanation: Internal error. Invalid link number or MERVA queue type encountered.

System Action: MQIA terminates.

User Response: Contact your local IBM representative.

CME4790E New message on MERVA queue could not be created [link *Link*, queue name *QueueName*, MERVA error *Code*].

Explanation: The creation of a new message on the denoted MERVA queue failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4793E New message on MERVA send queue could not be created [link *Link*, queue name *QueueName*, MERVA error *Code*].

Explanation: The creation of a new message on the denoted MERVA send queue failed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and resolve the problem.

CME4796E Error while checking for MQSeries dead-letter queue existence [queue manager *QueueManager*, completion *Code*, reason *Code*].

Explanation: An error occurred when MQIA tried to open the MQSeries dead-letter queue.

System Action: MQIA terminates.

User Response: Verify that the default MQSeries dead-letter queue exists and that the queue is not get and put inhibited.

CME4798E Error while trying to start MQSeries queue manager [queue manager *QueueManager*, OS Error *OS-Code*].

Explanation: MQIA tried to start the MQSeries queue manager and failed.

System Action: MQIA terminates.

User Response: Verify the queue manager name, check the OS error code and resolve the problem.

CME4802E Message could not be put on MQSeries initiation queue [queue name *QueueName*, completion *Code*, reason *Code*].

Explanation: Putting a message on the denoted MQSeries initiation queue failed.

System Action: MQIA terminates.

User Response: Verify the completion and reason code, and resolve the problem.

CME4805E **Base queue name of MQSeries alias queue could not be inquired** [*SectionType*, **queue name** *Queue*, **completion** *Code*, **reason** *Code*].

Explanation: The base queue name of an MQSeries alias queue could not be retrieved.

System Action: MQIA terminates.

User Response: Verify completion and reason code, resolve the problem, and restart MQIA.

CME4808W **Message on MERVA queue is in use** [*Section Type*, **queue name** *QueueName*].

Explanation: The message on the denoted MERVA queue is currently in use by another user.

System Action: MQIA continues processing. If MERVA recovery is enabled, MQIA tries to process the locked message later. If MERVA recovery is not enabled, MQIA tries to process the message during the next MQIA startup.

User Response: Check whether another MERVA API process is working on the appropriate queue. If not, restart MERVA and MQIA.

CME4809W **No message found on MERVA queue** [*Section Type*, **queue name** *QueueName*].

Explanation: The denoted MERVA queue is currently empty.

System Action: MQIA continues processing and waits for new messages.

User Response: None.

CME4812E **Error during update of MERVA-related field** [*Section Type*, **queue name** *QueueName*, **MRN** *MRN*, **MERVA error** *Code*].

Explanation: A MERVA-related field in the message could not be updated.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and the MQIA log file for possible reasons.

CME4813E **Message put on MERVA send queue failed, error during writing fields values to MERVA message space** [**link** *Link*, **queue name** *QueueName*, **MRN** *MRN*, **MERVA error** *Code*].

Explanation: Putting a message on the denoted MERVA send queue failed while filling the MERVA-related data fields.

System Action: MQIA terminates.

User Response: Verify the error code and the MQIA log file for possible reasons.

CME4815E **Error removing message from MERVA queue** [**link** *Link*, **queue** *QueueName*, **MERVA error** *Code*].

Explanation: MERVA message could not be deleted from denoted queue.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and the MQIA log file for possible reasons and resolve the problem.

CME4817E **Message check on MERVA send queue failed** [**link** *Link*, **queue name** *QueueName*, **MERVA error** *Code*].

Explanation: The current MERVA message does not comply to the SWIFT or Telex rules. The check depends on whether it is a SWIFT or Telex message.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and the MQIA log file for possible reasons.

CME4819E **Not all messages could be put on MQSeries initiation queue** [**queue name** *QueueName*, **number of outstanding messages** *OutstandingMsg*, **completion** *Code*, **reason** *Code*].

Explanation: An error occurred while putting messages on the MQSeries initiation queue.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and the MQIA log file for possible reasons.

CME4821E **MERVA message check failed** [*Section Type*, **queue name** *QueueName*, **MRN** *MRN*, **MERVA error** *Code*].

Explanation: The MERVA message with the denoted message reference number MRN does not comply with the SWIFT or Telex rules. The check depends on whether it is a SWIFT or Telex message.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and the MQIA log file for possible reasons.

CME4823E **Error opening MQSeries queue manager** [**queue manager group** *QueueManagers*, **completion** *CompletionCode*, **reason** *ReasonCode*].

Explanation: MQIA could not open the queue manager for retrieving the queue manager name.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4824E   **Error retrieving MQSeries queue manager name [queue manager group** *QueueManagers*, **completion** *CompletionCode*, **reason** *ReasonCode*].

Explanation: MQIA could not retrieve the queue manager name.

System Action: MQIA terminates.

User Response: Verify the completion and reason code and resolve the problem.

CME4900E   **Transaction object could not be created.**

Explanation: The transaction object could not be created.

System Action: MQIA terminates.

User Response: Verify the MQIA log file for possible reasons.

CME4901E   **Invalid link number specified for MERVA-to-MQSeries transaction [link number** *Link*].

Explanation: Internal error. Invalid link number encountered while starting a new transaction.

System Action: MQIA terminates.

User Response: Verify the MQIA log file for possible reasons and contact your local IBM representative.

CME4917E   **Invalid link name or MERVA send queue name in transaction message [transaction queue** *QueueName*, **link name** *Link*, **MERVA send queue** *MERVASendQ*].

Explanation: Internal error. Invalid link name or MERVA send queue encountered in transaction message.

System Action: Message is put in the MQSeries dead-letter queue and MQIA terminates.

User Response: Verify MQIA log file for possible reasons and contact your local IBM representative.

CME4927E   **No corresponding request or datagram found in the MQSeries reference queue, put received** *MessageType* **in dead-letter queue [link** *Link*, **MQSeries receive queue** *QueueName*, **reference message ID** x*MsgId*, **incoming message ID** x*MsgId*].

Explanation: No corresponding request or datagram found in MQSeries reference queue to correlate the received reply or report.

System Action: The received message is put in the MQSeries dead-letter queue with an appropriate reason code in the dead-letter queue header.

User Response: Check your system settings and the remote system. Verify the MQSeries network configuration.

CME4932E   **No corresponding request or datagram found in the MERVA wait queue, put received** *MessageType* **in dead-letter queue [link** *Link*, **MERVA wait queue** *QueueName*, **reference message ID** x*MsgId*, **incoming message ID** x*MsgId*].

Explanation: No corresponding request or datagram found in the MERVA wait queue to correlate the received reply or report.

System Action: The received message is put in the MQSeries dead-letter queue with an appropriate reason code in the dead-letter queue header.

User Response: Check your system settings and the remote system. Verify the MQSeries network configuration.

CME4935E   **Received request on MQSeries receive queue already on MQSeries reference queue [link** *Link*, **MQSeries receive queue** *QueueName*, **MERVA** *QueueType* **receive queue** *QueueName*, **message ID** x*MsgId*].

Explanation: A request with the denoted message ID is already in the system.

System Action: The request is put in the MQSeries dead-letter queue with the appropriate reason code in the dead-letter queue header.

User Response: Check the remote system and the MQSeries network configuration. Using multiple links it is possible that the same message ID was created by another remote system.

CME4939E   **Invalid link number specified for MQSeries-to-MERVA transaction [link number** *Link*].

Explanation: Internal error. Invalid link number encountered while starting a new transaction.

System Action: MQIA terminates.

User Response: Verify MQIA log file for possible reasons and contact your local IBM representative.

CME4960E **Semaphore call failed in order to indicate the arrival of a message in a MERVA send queue [link** *Link*, **queue name** *QueueName*, **MERVA error** *Code*].

Explanation: The state of the MERVA send queue semaphore of the denoted queue could not be changed.

System Action: MQIA terminates.

User Response: Verify the MERVA error code and MQIA log file for possible reasons and resolve the problem.

CME4961E **Checking of MERVA send queue failed [link** *Link*, **queue name** *QueueName*].

Explanation: The MERVA send queue could not be checked for new messages.

System Action: MQIA terminates.

User Response: Verify MQIA log file for possible reasons and resolve the problem.

CME4966E **Invalid message format in MQ receive queue, put received** *MsgType* **in dead-letter queue [link** *LinkName*, **MQ receive queue** *QueueName*, **incoming CorrelID** x*MQMsgId*, **incoming MsgID** x*MQMsgId*, **DLQ reason** *Code*].

Explanation: The incoming message in the MQSeries receive queue contains an invalid format.

System Action: The message cannot be processed and is put in the MQSeries dead-letter queue.

User Response: Depending on the dead-letter queue reason code check the remote queue manager or the application on the remote site.

CME4967E **Invalid MQSeries message type in the MQ receive queue [link** *Link*, **MQ receive queue** *QueueName*, **message type** *MsgType*].

Explanation: The incoming message in the MQSeries receive queue has an MQSeries message type which is not supported by MQIA.

System Action: The message cannot be processed and is put in the MQSeries dead-letter queue.

User Response: Check the application on the remote site.

CME4968E **Move invalid MQSeries message from MQSeries receive queue to dead-letter queue [link** *Link*, **MQ receive queue** *QueueName*, **message ID** x*MsgId*].

Explanation: The incoming message was either too big or a conversion problem occurred.

System Action: The message cannot be processed and is put in the MQSeries dead-letter queue. MQIA continues processing.

User Response: Check the MQIA log file for possible reasons, for example, the completion and reason code of the previous MQSeries message get operation.

CME5006E **Memory allocation failure!**

Explanation: The required memory could not be allocated.

System Action: The system is running out of memory. MQIA is terminating.

User Response: Verify your system resources and resolve the problem. Close some applications or reboot the system.

CME5008E **Unknown argument <***Arg***> specified!**

Explanation: MQIA was started with an unknown argument.

System Action: MQIA terminates.

User Response: Check your arguments if MQIA was started from command line.

CME5010E **Argument** *Arg* **was specified more than once!**

Explanation: An argument was specified more than once and is allowed only once.

System Action: MQIA terminates.

User Response: Specify the required argument once if you start MQIA from the command line.

CME5020E **Unable to install MQIA service. MERVA installation path not found.**

Explanation: The MERVA installation path could not be read from the Windows NT registry.

System Action: The system is not able to install the MQIA service.

User Response: Check your MERVA installation.

**CME5021E Unable to install MQIA service.
Configuration file** *FileName* **not found.**

Explanation: The specified file was not found by the service installation utility.

System Action: The system is not able to install the MQIA service.

User Response: In the Create service - Settings window enter an existing configuration file in the field MQIA profile name.

**CME5022E Unable to install MQIA service. MQIA
program** *ProgName* **not found.**

Explanation: The MQIA program was not found by the service installation utility.

System Action: The system is not able to install the MQIA service.

User Response: Verify the MERVA installation log for errors and check the 'bin' directory of MERVA for the existence of the program cmemqat.exe. Reinstall MERVA if the program cmemqat.exe is not located in the 'bin' directory of MERVA.

CME5023E Invalid parameter entered.

Explanation: The service installation utility received an unknown parameter.

System Action: The utility terminates.

User Response: Use the MERVA Control Center.

**CME5025E Unable to install MQIA service. Invalid
file extension '***Extension***' specified for
MQIA configuration file name. It must
be '.ATN'.**

Explanation: The file extension of the specified configuration file is not valid.

System Action: The system is not able to install the MQIA service.

User Response: Enter a configuration file name with the extension .ATN or without an extension.

CME5030E Configuration file *FileName* **already
exists.**

Explanation: A file with the specified file name already exists.

System Action: The migration utility does not overwrite a file.

User Response: Verify the existing file. If it is not the already migrated configuration file, either delete or rename it.

CME5032E Configuration file *FileName* **not found.**

Explanation: The specified configuration file was not found.

System Action: The migration utility terminates.

User Response: Start the migration utility with an existing configuration file (extension .att).

Appendix G. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Deutschland
Informationssysteme GmbH
Department 3982
Pascalstrasse 100

95

70569 Stuttgart
Germany

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

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement or any equivalent agreement between us.

The following paragraph does apply to the US only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:
- Advanced Peer-to-Peer Networking
- AIX
- APPN
- C/370
- CICS
- CICS/ESA
- CICS/MVS
- CICS/VSE
- DB2
- Distributed Relational Database Architecture
- DRDA
- eNetwork
- IBM
- IMS/ESA
- Language Environment
- MQSeries

- MVS
- MVS/ESA
- MVS/XA
- OS/2
- OS/390
- RACF
- VSE/ESA
- VTAM

Workstation (AWS) and Directory Services Application (DSA) are trademarks of S.W.I.F.T., La Hulpe in Belgium.

Pentium is a trademark of Intel Corporation.

PC Direct is a trademark of Ziff Communications Company in the United States, other countries, or both, and is used by IBM Corporation under license.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Index

Special Characters

Readers' Comments — We'd Like to Hear from You

MERVA ESA Components
MQI Attachment
User's Guide
Version 4 Release 1

Publication No. SH12-6714-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you?   ☐ Yes   ☐ No

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.

Name

Address

Company or Organization

Phone No.

Readers' Comments — We'd Like to Hear from You

SH12-6714-00

IBM®

Cut or Fold
Along Line

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland Entwicklung GmbH
Information Development & User Centered Design
Dept. 0446
Schoenaicher Strasse 220
71032 Boeblingen
Germany

Readers' Comments — We'd Like to Hear from You

SH12-6714-00

Cut or Fold
Along Line

IBM ®

Program Number: 5648-B30

Java and all Java-based trademarks
and logos are trademarks of Sun
Microsystems, Inc. in the United States
and other countries.