MERVA for ESA

Installation Guide

Version 4 Release 1

SH12-6378-01

MERVA for ESA

Installation Guide

Version 4 Release 1

SH12-6378-01

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix E. Notices" on page 149.

Second Edition, May 2001

This edition applies to Version 4 Release 1 of IBM MERVA for ESA (5648-B29) and to all subsequent releases and modifications until otherwise indicated in new editions.

Changes to this edition are marked with a vertical bar.

Contents

About This Book

This book describes how to install and modify the IBM licensed program Message Entry and Routing with Interfaces to Various Applications for ESA Version 4 Release 1 (abbreviated to MERVA ESA in this book). MERVA ESA provides the following components, which you can install as required:

- Base Functions
- SWIFT Link
- Telex Link
- MERVA Link
- MERVA-to-MERVA Financial Message Transfer/ESA (FMT/ESA) using MERVA Link

This book contains detailed technical information about:

- Hardware, software, CICS, IMS, and storage requirements for MERVA ESA
- Installation and system generation of MERVA ESA network links
- Installation verification procedures and test commands
- Maintaining MERVA ESA using SMP/E and MSHP
- Telex Link protocol

The installation of Traffic Reconciliation is described in *MERVA for ESA Traffic Reconciliation*.

Prerequisites for Using This Book

Persons using this book should be familiar with the manual *MERVA for ESA Concepts and Components*, which describes the functions, services, and utilities supplied with MERVA, and explains the concepts of messages, queues, routing, message handling, and network links.

For the SWIFT Link, it is assumed that persons using this book are familiar with the contents of the *S.W.I.F.T. User Handbook*, which is published by the Society for Worldwide Interbank Financial Telecommunication s.c. (S.W.I.F.T.). For the Telex Link, readers should be familiar with the telex terminology as defined by their their local post and telecommunication authority (sometimes referred to as their PTT, for ″post, telegraph, telephone″).

Persons reading this book should be familiar with:

- MERVA ESA
- The operating system under which MERVA ESA is used: MVS/ESA™, OS/390®, or VSE/ESA™
- The data communication system IMS or CICS under which MERVA ESA is used

Note: The term *CICS* is used to refer to CICS/ESA®, CICS Transaction Server (CICS TS), and CICS/VSE®. The term *IMS* is used to refer to IMS/ESA®.

Summary of Changes

This edition reflects the following changes:

New information for systems using both MERVA Link USS and OS/390 UNIX-level security

See “Systems Using OS/390 UNIX-Level Security” on page 59 for details.

MERVA-MQI Attachment installation verification

MQSeries® for VSE/ESA now fully exploits COA and COD report message handling. Therefore, the installation of the MERVA-MQI Attachment under VSE can be verified in the same way as under MVS. See “Chapter 16. Verifying the Installation of MERVA-MQI Attachment” on page 89.

FMT/ESA can now use MERVA-MQI Attachment

Financial Message Transfer/ESA (FMT/ESA) can now use MERVA-MQI Attachment as well as MERVA Link ESA to transfer SWIFT messages between two MERVA ESA systems. See “Chapter 17. Verifying the Installation of FMT/ESA Using MERVA-MQI Attachment” on page 103.

Part 1. Installation Prerequisites

This part describes the basic installation requirements: the operating environment, system requirements, external storage requirements, and important features of the operating system and data communication system.

Chapter 1. Operating Environment

This chapter describes the machine and programming requirements for MERVA ESA.

Note: Where release levels are given in this chapter they also apply to later releases of the required programs, unless otherwise stated.

MERVA ESA Machine Requirements

Processors

MERVA ESA runs on any processor that supports the specified operating system environment. The required size of the processor depends on whether the operating system is MVS™ or VSE, and on whether the data communication system used to control the operation of MERVA ESA is MVS/CICS, VSE/CICS, or MVS/IMS.

Peripheral Equipment

To run MERVA ESA you need:

- At least one 3270-compatible display station, for example, 3278-2 or 3279-2.
- At least one IBM 3268 Printer or any printer operating in 3270 mode or SCS mode.
- A tape or cartridge drive for initial loading.
- At least one direct-access storage device (DASD) for data sets and libraries; as supported by the operating system.

Storage Requirements

Base Functions

When running in a CICS environment, MERVA ESA needs 2.4MB[1] plus an additional 170KB[2] for each additional user or printer task, and an additional 90KB for each additional DSLCXT task. The SWIFT Link needs 650KB in the CICS region or partition. The required storage must be located in a CICS region or partition.

When running in an IMS environment, MERVA ESA needs 1.6MB in the BMP region and 800KB in the MPP region; an additional 500KB must be allocated per MPP region or in the LPA. The SWIFT Link needs 650KB in the BMP region, and 600KB in either each MPP region or in the LPA.

Lines to the SWIFT Network

For X.25 communication links to the SWIFT network, MERVA ESA uses the product MERVA Extended Connectivity. For the SNA session to MERVA Extended Connectivity, additional storage is needed. The additional storage is only needed when the lines are active. The storage is allocated in the region where the MERVA ESA nucleus is running. When running under CICS, this storage is not part of the CICS storage, but obtained directly from the operating system.

Independent of the number of active lines, the following storage is needed:

[1] One megabyte (1MB) equals 1 048 576 bytes.

[2] One kilobyte (1KB) equals 1024 bytes.

- For MVS and VSE, the reentrant line module needs 192KB above the 16MB line.
- For VSE, the network link module needs an additional 12KB below the 16MB line.

For each active line, the following storage is needed:

- 52KB as basic working storage
- 205KB for the transport layer (this storage size is customizable)
- 23KB for the CBT
- 11KB for each login of an LT
- 11KB for each select of an LT

This storage is all allocated above the line with the exception of about 10KB below the 16MB line.

The following formulas determine the total storage required (in KB):

- For MVS: 192 + (280 × number_of_lines) + (22 × number_of_LTs)
- For VSE: 204 + (280 × number_of_lines) + (22 × number_of_LTs)

Batch Programs and Utilities

The batch programs and the utilities need 4MB.

MVS Interregion Communication

The MERVA ESA SVC allocates buffer storage in the ECSA (subpools 231 and 241). MERVA ESA needs about 1KB in the non-fetch protected storage (subpool 241).

For each ICB used for interregion communication one parameter list buffer and one data buffer is used which are allocated in the fetch-protected storage (subpool 231). The basic size of these buffers is specified by parameters NICPL and NICBUF in customization parameter module DSLPRM. When dynamic buffers are used, for example, the MAXBUF parameter specification is greater than the NICBUF specification, more space is needed when larger messages are processed. The data buffer can grow up to the size specified in the MAXBUF parameter.

Additional Machine Requirements for the SWIFT Link

The connection to the SWIFT network using X.25 requires one communication controller of type 3720, 3725, or 3745. For these controllers:

- Physical connections to the SWIFT network require LIC type 1 (EIA RS232 or CCITT V.24; max. 19200 bps) or LIC type 3 (CCITT V.35).
- Switched (PSTN) connections to the SWIFT network require a modem that supports manual dialing (for 3720, 3725, or 3745), DTR dialing (for 3720, 3725, or 3745), or V.25 bis dialing (for 3745 only). A modem capable of V.25 bis dialing lets MERVA ESA control automatic dialing and is recommended.

A direct-access storage device compatible with the operating system is needed to store SWIFT Link data sets.

Additional Machine Requirements for the MERVA Link

The connection between the host processor and another MERVA ESA system requires one of the following network control facilities:

- An IBM 37xx Communication Controller
- An Integrated Communication Adapter (ICA)

Workstation Machine Requirements for MERVA ESA

The workstation machine requirements for using the following functions are described in the current *MERVA for ESA Licensed Program Specifications*:

- S.W.I.F.T. USE functions
- Telex network connection via workstation
- MERVA-to-MERVA connection between MERVA ESA and MERVA Link partners

Additional Machine Requirements for the Telex Link Via a Fault-Tolerant System

You can communicate with the public telex network via a fault-tolerant front end and a PTT [3] approved TelexBox, such as the Intercope TelexBox. For details refer to the *Telex Interface Program: Program Description and Operations Manual.*

The connection to the public telex network requires the following:

- 1 to 32 telex lines for the Telex Link via a fault-tolerant system.
- At least one PTT-approved TelexBox such as the Intercope TelexBox, connected via a serial adapter with up to a maximum of 8 telex boxes covering a maximum of 4 telex lines each.
- Approved modems, if required by local PTT regulations or carrier.
- A printer per TelexBox, if required by local PTT regulations.

MERVA ESA Program Requirements

System Environment

MERVA ESA requires the following control programs and subsystems (subsequent releases can be used, unless otherwise stated):

- One of the following operating systems:
  - VSE/ESA Version 2 Release 3 (5690-VSE)
  - OS/390 Version 2 Release 4 (5647-A01)
- One of the following database and communications systems:
  - CICS/VSE Version 2 Release 3 (5686-026)
  - CICS/ESA Version 4 (5685-018)
  - CICS Transaction Server Version 1 (5655-147)
  - IMS/ESA Version 5 (5695-176)
- For CICS/VSE:
  - The HL Assembler for VSE (5696-234)
- When using MERVA-MQI Attachment, one of the following:
  - MQSeries for MVS/ESA Version 1 Release 2 (5695-137)
  - MQSeries for VSE/ESA Version 2 Release 1 (5686-A06)
- When using intertask communication via MQSeries or interservice communication via MQSeries:
  - MQSeries for MVS/ESA Version 1 Release 2 (5695-137)
- When using queue management using DB2®, one of the following:
  - DB2 for OS/390 Version 5 Release 1 (5655-DB2)
  - DB2 Server for VSE & VM Version 5 Release 1 (5648-158)

[3] National Post and Telecommunication (post, telegraph and telephone).

- For MVS, when using the MERVA ESA interface to an external security manager (ESM), and when RACF® is that ESM, one of the following:
  - RACF Version 1 Release 9 (5740-XXH)
  - RACF Version 2 Release 1 (5695-039)
- For VSE, when using the MERVA ESA interface to a security manager, both of the following:
  - VSE/ESA Version 2 Release 4 (5690-VSE)
  - CICS Transaction Server for VSE/ESA Version 1 Release 1 (5648-054)

SWIFT Network Connection

The connection to the SWIFT network is possible via X.25.

One of the following programs is needed:

- MERVA Extended Connectivity for MVS/ESA Version 3 Release 2 (5655-110)
- MERVA Extended Connectivity for VSE/ESA Version 3 Release 2 (5686-080)

For the 37xx Communications Controller used by MERVA Extended Connectivity, one of the following NCPs is needed:

- ACF/NCP Version 4 Release 3.1 (5668-854), for MVS or VSE
- ACF/NCP Version 5 Release 4 (5668-738), for MVS or VSE
- ACF/NCP Version 6 Release 1, 2, or 3 (5688-231), for MVS only
- ACF/NCP Version 7 Release 1, 2, 3, or 4 (5648-063), for MVS or VSE

Workstation

For the S.W.I.F.T. USE functions or the telex network connection you need a MERVA-to-MERVA connection between MERVA ESA and MERVA USE Workstation.

The requirements for a MERVA-to-MERVA connection between MERVA ESA and the workstation are described in *MERVA for ESA Licensed Program Specifications*.

Chapter 2. System Requirements of MERVA ESA

This chapter describes the features required by MERVA ESA from the operating and data communication systems.

MVS Requirements

MERVA ESA provides facilities for:

- Intertask communication between MERVA ESA programs using an SVC
- Intertask communication between MERVA ESA programs using APPC/MVS
- Intertask communication between MERVA ESA programs using MQSeries
- Interservice communication between MERVA ESA instances on a sysplex
- Using an external security manager such as RACF

Intertask communication is called interregion communication when using the SVC.

SVC for Interregion and Interservice Communication

MERVA ESA interregion and interservice communication require a type-3 SVC routine in the MVS system. With MERVA ESA, the following SVC programs are delivered:

DSLNICPM This program is compatible with the MERVA ESA V3.3 program of the same name. Although V4.1 of MERVA ESA can run using the SVC of V3.3 (for example, during an initial test run), to simplify maintenance and to be sure you have all the latest PTFs, you should install and use this new SVC program.

DSLXSVCX This program includes the interregion communication services of DSLNICPM, but is extended with the function used by interservice communication. DSLXSVCX is only required when several MERVA ESA systems are to run in a sysplex.

Whether you use interregion communication (and hence the SVC) depends in part on whether you use CICS or IMS:

- In CICS, interregion communication is optional and only used for the following services:
  - The batch programs DSLSDI, DSLSDO, and DSLSDY
  - Batch user applications using DSLAPI
  - End-user communication from one CICS system without a started MERVA ESA to another CICS system with a started MERVA ESA
- In IMS, interregion communication is always required

The SVC number must be provided by means of the SVCTABLE macro for stage 1 of the MVS system generation as shown below:

```
SVCTABLE  SVC nnn
```

where *nnn* is the SVC number. This number is also specified in the SVC parameter of the customizing parameter module DSLPRM.

Install:

- DSLNICPM, if you use only interregion communication.

- DSLXSVCX, if you use interservice communication.

During installation of MERVA ESA, DSLNICPM or DSLXSVCX is copied into the SYS1.LPALIB with the correct module name (IGC00...) for an SVC routine.

Interregion Communication

The MERVA ESA interregion communication program DSLNICPM uses one of the following:

MVS subsystem entry
You define as many entries as there are MERVA ESA systems running in this MVS when using the subsystem entry.

CVTUSER entry of the MVS Communication Vector Table (CVT)
The CVTUSER entry is used to point to an extension table to allow several MERVA ESA installations and other applications to use the CVTUSER field. Each of the MERVA ESA installations and other applications use one or more fullwords in this extension table for their purposes. If necessary, the other applications must be changed to follow this rule. The first program that starts after an IPL of MVS creates the table when it finds that the CVTUSER is zero, and places the table address in the CVTUSER field.

The CVT extension table provided by MERVA ESA has the following layout:

- A length of 168 (including the identification tag)
- An offset of 160 to the identification tag
- An identification tag 'CVTEXTID'

To conform with MVS system integrity rules, this layout is hard-coded in the DSLNICPM (SVC) program. Other applications that create the CVT extension table must provide the same layout.

The DSLNICPM program follows the rules of MVS system integrity. DSLNICPM uses, for each MERVA ESA to be run, one fullword of the CVT extension table for the address of the MERVA ESA Interregion Communication Area DSLICA. The parameter CVTEXTO of the DSLPARM macroinstruction (which is used in the MERVA ESA customizing parameter module DSLPRM) defines which fullword of the CVT extension table is used by the particular MERVA ESA.

This also means that different MERVA ESA installations running in the same MVS system must have different DSLPRM modules with different CVTEXTO parameters. Storage of the CVT extension table not used by MERVA ESA can be used by other applications.

For example, to start two MERVA ESA installations, you code:

- CVTEXTO=0 in the DSLPRM module for the first MERVA ESA
- CVTEXTO=4 in the DSLPRM module for the second MERVA ESA

The storage from offset 8 to offset 156 can be used by other applications.

If you are using subsystem entries, specify MVSSS=YES in the DSLPRM module. Each MERVA ESA running in the same MVS system must have a different DSLPRM module with a different DSLID parameter.

The subsystem entries are specified in the SYS1.PARMLIB member with the name **IEFSSN*xx*** , where *xx* is a suffix used in your MVS installations. The name of the

DSLID parameter is specified in columns 1 to 4, followed by at least one blank and, in some cases, comments. For example, if DSLID=MERV were specified in DSLPRM, it would contain the entry:

```
MERV           Entry for MERVA ESA Interregion Communication
```

Intertask Communication Using APPC/MVS

The MERVA ESA nucleus task server program DSLNTSA registers with APPC/MVS to serve the inbound requests from MERVA ESA clients. When running in a BMP, IMS prohibits the use of a NOSCHED LU. In this case the program DSLNTSA must run in its own region.

The program DSLNTSAB must be started to serve as an interface between the MERVA ESA APPC/MVS server DSLNTSA and the MERVA ESA nucleus running in the BMP. The communication between DSLNTSAB and the MERVA ESA nucleus is via the traditional MERVA ESA interregion communication using the SVC.

After DSLNTSAB is started it runs as long as MERVA ESA is active. When MERVA ESA is no longer active, or the interregion communication to the MERVA ESA nucleus is not possible, the program DSLNTSAB terminates.

It is possible to start more than one DSLNTSAB batch program serving the same or a different TP names.

The command DICB can be used to monitor the status of the APPC/MVS servers started with the program DSLNTSAB. A DSLNTSAB server can be stopped by an authorized MERVA ESA operator using the DICB command. For a detailed description of the DICB command, refer to the *MERVA for ESA Diagnosis Guide*.

The JCL for the server DSLNTSAB must include the MERVA ESA load module library. Figure 1 shows the JCL required to run the APPC/MVS server DSLNTSAB.

```
//.......  JOB .........
//RUN      EXEC PGM=DSLNTSAB,REGION=4M,PARM='parm1,parm2,parm3,parm4'
//STEPLIB  DD DSN=merva.SDSLLODB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//DSLSNAP  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

Figure 1. Running the MERVA ESA APPC/MVS Server Batch Program

The PARM= statements specify the parameters used by the batch APPC/MVS server DSLNTSAB. Some parameters are taken from the DSLPRM customization module installed in the load library.

In the JCL, the lowercase parameters have the following meaning:

merva.SDSLLODB
The library containing the MERVA ESA load modules. The module DSLPRM in the load library must not be loaded into write-protected storage; that means it must not reside in an LPALIB or APF authorized library.

parm1 The maximum number of sessions supported by this APPC/MVS server. A number between 1 and 254 can be specified. The default is 5.

parm2 The name of the APPC/MVS server as it appears in the 'DICB SERVERS' command response. If two or more MERVA ESA batch APPC/MVS servers are started, they should have different names.

parm3 The TP name used for this server. When the parameter is omitted the TP name specified in the ITCASRV parameter in the DSLPRM customization module is used.

parm4 Specify Y to indicate that the MERVA ESA batch APPC/MVS should retry to connect to MERVA ESA after the MERVA ESA nucleus has terminated. After an interval of 1 minute, the server tries to reconnect to MERVA ESA. Up to 10 attempts at connecting to MERVA ESA are made until the batch APPC/MVS server terminates.

Intertask Communication Using MQSeries

The MERVA ESA nucleus task server program DSLNTSM connects to MQSeries and processes the inbound requests from MERVA ESA requesters. The parameter ITCMRCV of the parameter module DSLPRM specifies the MQSeries receive queue processed by DSLNTSM.

The command DICB can be used to monitor the status of the intertask communication servers. For a detailed description of the DICB command, refer to the *MERVA for ESA Diagnosis Guide*.

Intertask communication needs a set of MQSeries Message Queue definitions. Sample Message Queue definitions are in the copy books DSLNITC1 for the MERVA ESA nucleus task server and DSLNITC2 for client programs. You can use the MQSeries utility program CSQUTIL to issue the commands from a sequential data set (see “MQSeries Modifications” on page 11). Before you run the DEFINE commands change the names for your environment. The names assigned to the queues must also be defined in the ITCMxxxx parameters in module DSLPRM.

Interservice Communication Using MQSeries

This facility allows multiple MERVA ESA instances to run on different systems with shared MERVA ESA resources. Interservice communication is performed in each MERVA ESA instance by the MQSeries DSLNMQS. Interservice communication can be enabled by specifying the ISCMxxxx parameters in the parameter module DSLPRM.

DSLNMQS connects a MERVA ESA instance to MQSeries and establishes the connection between the MQI queues. Each MQSeries needs one local receive queue, one local reply-to queue, and one send queue.

The interservice communication needs a set of MQSeries Message Queue definitions. Sample Message Queue definitions are in the copy book DSLNISC1. You can use the MQSeries utility program CSQUTIL to issue the commands from a sequential data set (see “MQSeries Modifications” on page 11). Before you run the DEFINE commands change the names for your environment. The names assigned to the queues must also be defined in the ISCMxxxx parameters in module DSLPRM.

One of the MERVA ESA instances must be defined as the primary MERVA ESA instance. Only this MERVA ESA instance accepts MERVA ESA requests and maintains the connection to your application.

The QNAME parameter in the Nucleus Server Table (DSLNSVT) entry defines to which MERVA ESA instance the service belongs.

To enable the primary MERVA ESA instance to automatically start the secondary MERVA ESA instance (DSLPRM ISCSTART=AUTO) you must provide a cataloged procedure with the name you have specified in DSLNSVT (STCNAME=xxxxxxxx). A sample is provided with MERVA ESA (DSLNISCP). Modify the sample according to your environment before you copy it to the procedure library.

For a detailed description of how to customize modules DSLPRM and DSLNSVT refer to the *MERVA for ESA Customization Guide* and *MERVA for ESA Macro Reference*.

MQSeries Modifications

You can use the MQSeries utility program CSQUTIL to issue MQSeries commands from a sequential data set. Sample copy books for creating the necessary MQSeries objects are delivered with MERVA ESA, and are described in the MERVA ESA program directory. In these copy books, the parameters of the MQSeries DEFINE command that have to be checked and modified before the utility can be run are indicated. For more information about the MQSeries DEFINE command, refer to *MQSeries Command Reference*.

Using an External Security Manager

In order to use the services of an external security manager like RACF, MERVA ESA requires another type-3 SVC routine in the MVS system if the IMS subsystem is used.

For CICS/ESA Version 4 Release 1 or higher, you can decide whether to use the provided CICS service or the type-3 SVC. If you want to use the SVC, uncomment the instruction that follows the EXEC CICS call in the MERVA ESA End User Driver user exit DSLEU004:

```
         EXEC CICS INQUIRE SYSTEM RELEASE(U004CREL) NOHANDLE.
******** MVC   U004CREL,=C'0000'   ENFORCE USAGE OF THE SVC
```

Install DSLEU004 using SMP/E. DSLEU004 is link-edited to the MERVA ESA end user driver DSLEUD. Install the type-3 SVC.

The SVC number must be provided by means of the SVCTABLE macro for stage 1 of the MVS system generation as shown below:

```
SVCTABLE  SVC nnn
```

where *nnn* is the SVC number. This number is also specified in the RACFSVC parameter of the customizing parameter module DSLPRM. During the installation of MERVA ESA, the program DSLEUSVC is copied into the SYS1.LPALIB with the correct module name (IGC00...) for an SVC routine.

VSE Requirements

MERVA ESA uses the XPCC facility of VSE for interpartition communication. If several MERVA ESA systems run in the same VSE system, each must have a different DSLPRM with a different DSLID parameter.

Part 2. Installation of MERVA ESA and Its Components

This part describes:

- The layout of the machine-readable material for MERVA ESA and its components:
  - Base Functions
  - SWIFT Link
  - Telex Link
  - MERVA Link
  - MERVA-to-MERVA Financial Message Transfer/ESA using MERVA Link

  You should read this part together with the program directory for MERVA ESA.
- The space required for the data sets of MERVA ESA and its components.
- The changes and additions to the tables and control information of CICS and IMS that are necessary to run MERVA ESA.

Chapter 3. Installing the Base Functions

MERVA ESA is available for VSE running under CICS, and for MVS. Under MVS, the MERVA ESA programs can run in one or more CICS regions, in IMS MPP regions, in a BMP region, or as batch programs. Some of the programs run only in one of the supported environments, others can run in more than one environment.

With the distribution material you receive the appropriate program directory containing exact instructions on how to install MERVA ESA.

The layout of the tapes or cartridges differs between MVS and VSE, however they all contain:

- Complete sets of the JCL needed to install the product
- Object code of the product
- Source code of macroinstructions needed for customizing
- Sample programs, sample user exits, message control blocks (MCBs), and MERVA ESA tables

For MVS, the object code is delivered in three target libraries (executables):

- One containing the modules needed for CICS
- One containing the modules needed for IMS
- One containing the common modules and the modules needed for a batch environment

MERVA ESA is a complete pregenerated system that, after successful installation, can be run and tested. Some of the MERVA ESA tables have to be adapted to your installation environment. Refer to "Chapter 10. Generating MERVA ESA Tables" on page 61 to find out how to make the necessary changes and how the MERVA ESA Table Generation process works.

To implement the system according to the needs of your organization and operating environment, refer to the*MERVA for ESA Customization Guide*, the *MERVA for ESA Operations Guide*, the *MERVA for ESA Macro Reference*, and *MERVA for ESA Concepts and Components*.

Calculating the Space for the MERVA ESA Data Sets

This section shows methods for calculating space for allocating the MERVA ESA data sets.

User File

Each user file record is 1300 bytes long. You determine how many users will work with MERVA ESA and specify the number of records required in the VSAM cluster definition for the user file. MERVA ESA can use a scrambling algorithm to protect user file records. To activate the scrambling algorithm, specify the parameter EXUMASK=YES in the customization parameter module DSLPRM.

Note: You can use your MERVA ESA V3.3 user file in MERVA ESA V4.1.

Queue Data Set

If you use VSAM rather than DB2 for queue management, the number of records to be allocated for the queue data set is determined as follows:

- Determine how many messages for external networks (for example, SWIFT) must be stored in the queue data set in one day.
- Determine for how many days the messages should be held in the queue data set.
- Determine the average message length in the MERVA ESA queue format. For the SWIFT Link the average length of a SWIFT message in the MERVA ESA queue format is 3000 bytes. For large messages, only status information with an average length of about 400 bytes is stored in the queue data set, the message data is stored in the large message cluster (LMC).
- The record size of the queue data set is 32760 bytes. With this number, you can calculate how many messages fit into one record:
  - About 10 SWIFT messages of the SWIFT Link
  - The status information of about 80 large messages
  - About 250 session keys when SWIFT secure login/select (SLS) with session key preload is used.

For example, for 2000 messages each day with messages held in the queue data set for 5 days:

```
2000 messages * 5 days
---------------------- = 1000 records for the queue data set
10 messages per record
```

The maximum number of records is the number of system records plus the number of data records. The number of system blocks depends on your NQE definition in DSLPRM and the key definitions in the function table (DSLFNTT). The utility DSLQDSUT informs you about the number of system records needed after a FORMAT or MODIFY function. The maximum number of data records is 65512 no matter how many system records are needed. If you want to make use of the maximum number of records, you run DSLQDSUT for FORMAT to find out the number of system records, add 65512 and define the queue data set cluster with this number of records, and run DSLQDSUT for FORMAT again. If you define the queue data set cluster for more records than DSLQMGT can handle, DSLQDSUT will limit it automatically to the maximum.

The machine-readable material for MERVA ESA contains the JCL for the VSAM cluster definition and the formatting of the queue data set. If you want to use the duplicate queue data set feature of MERVA ESA, you must define two queue data sets and must format them both.

Note: To use your MERVA ESA V3.3 queue data set in MERVA ESA V4.1, you can run the DSLQDSUT utility with the following parameters:

- FORMAT, to format your MERVA ESA V3.3 queue data set for MERVA ESA V4.1 if you need not keep messages.
- MODIFY, to copy all messages from your MERVA ESA V3.3 queue data set into your MERVA ESA V4.1 queue data set.

Refer to *MERVA for ESA Operations Guide* for details about using DSLQDSUT.

You can migrate the queue data set to the Queue Management DB2 tables using the MERVA ESA V4 unload and load utilities DSLSDUR and DSLSDLR. These utilities will not run under MERVA ESA V3.3. They are described in *MERVA for ESA Operations Guide*.

Journal Data Set

The MERVA ESA journal data sets are used to store a variety of information (see the description of the journal in *MERVA for ESA Concepts and Components*).

There are two journal data sets in MERVA ESA, identified by the letter A and B. Only one data set is active (open) at a time. If the active data set becomes full during processing, MERVA ESA can switch to the alternate data set and continue processing without interrupting applications.

The use of two data sets and the ability to switch between the data sets during processing allows the use of an archiving procedure without terminating and restarting MERVA ESA.

The archived journal data set can be reset to an initial state and reused. The journal data sets must be defined with the appropriate DEFINE CLUSTER parameter in the allocation job to allow the reset. Refer to *MERVA for ESA Operations Guide* for details. A sample job stream provided by MERVA ESA outlines a procedure for daily archiving of the journal data sets without interrupting MERVA ESA processing.

The size of the journal data sets should be verified in a test system, and adjusted during production. An easy way to calculate the space for the journal data set is:

- Determine how many messages for external networks (for example, SWIFT) must be stored in the journal in one day.
- Determine for how many days the messages should be held in the journal.
- The number of all other records written to the journal besides messages for external networks is about the same as the number of records for external networks.
- Determine the average message length. With MERVA ESA alone or together with the SWIFT Link, the average record length in the journal is 2000 bytes. If you have other programs in your installation that use the MERVA ESA journal, you must determine if they use journal records with a higher or lower average record length, and use a value different from 2000 bytes if necessary.

For example, for 2000 messages each day and messages to be held in the queue data set for 5 days:

```
2000 messages * 5 days * 2 (for other journal records) = 20000 journal records.
```

For the MERVA ESA trace facilities, this calculation may have to be modified. The values in the following list show how much more space is required when traces are used. On the other hand, a journal filled with traces will most probably be kept for only one day or one test, so that the same journal data set can be used.

- If the MERVA ESA routing trace facility is used, add 1 to the factor 2 in the formula above, for example, 2 + 1 = 3.
- If the MERVA ESA queue trace facility is used, add 2 to the factor 2 in the formula above, for example, 2 + 2 = 4.
- If the MERVA ESA debugging trace facility is used, add 4 to the factor 2 in the formula above, for example, 2 + 4 = 6.

Note: You can use your MERVA ESA V3.3 journal data sets in MERVA ESA V4.1, provided that the 4-digit year format for the journal record header was used. In MERVA ESA V4.1 the 4-digit year format for the journal record header is enforced. You are recommended to allocate journal data sets with a record length of at least 16000 bytes when large SWIFT messages up to 10KB are to be processed with MERVA ESA.

MERVA ESA Nicknames File

The MERVA ESA nicknames file is optional. If nicknames expansion is defined in the MERVA ESA function table, the file must be defined and filled.

Determine the number of nicknames and use this number in the VSAM cluster definition of the MERVA ESA Nicknames File.

The machine-readable material for MERVA ESA contains the JCL for the VSAM cluster definition and the initialization of the Nicknames File.

Note: You can use your MERVA ESA V3.3 Nicknames File in MERVA ESA V4.1.

SPA File (IMS only)

The SPA file of MERVA ESA is used in IMS only. Its purpose is to allow a scratchpad area for the MERVA ESA End-User Driver larger than 32KB, the maximum size IMS can handle. IMS has to process a SPA of only 320 bytes; all other storage is saved in the SPA file of MERVA ESA.

The SPA for each end-user consists of up to four parts:

- The permanent storage. The size of the permanent storage is calculated by adding together the following:
  - The size of the End-User Driver permanent storage (5.5KB)
  - The size of the SPA for user exits (USERSTO parameter of the DSLPARM macro)
  - The size of MFS permanent storage and the retype buffer (the first and the third subparameters of MFSSTOR of the DSLPARM macro)
  - The size of the load table for the Message Control Blocks (the value of the MCBNUM parameter of the DSLPARM macro multiplied by 20)
  - The maximum of the permanent storage sizes of each function program (limited to 4KB)

  When using the default values during customization the total size of this permanent storage is 17268 bytes.
- The MERVA ESA tokenized form (TOF). Its size is determined by the TOFSIZE parameter of the DSLPARM macro. When a TOFSIZE larger than 32760 is specified the size is reduced to 32760.
- The MERVA ESA LDS (logical data stream). Its size is determined by the NICBUF parameter of the DSLPARM macro. When a NICBUF larger than 32760 is specified the size is reduced to 32760.
- The dynamic data TOF. The second subparameter of TOFSIZE specifies the amount of storage by which the TOF may grow dynamically when large messages are processed. The maximum size of this area is MAXBUF−TOFSIZE.

The number of users defined in the USER parameter of the DSLPARM macro, multiplied by 3, plus 1, gives the minimum number of records that must be available in the SPA File. For example, for 20 users the required minimum number of records is

```
20 users * 3 records + 1 index record  = 61 records.
```

For each end-user session three SPA file records are allocated in the SPA file. When the working storage to be saved between conversation steps does not fit into these three SPA file records, SPA file extension records are used. This can only happen when a message with an internal size larger than 32KB is processed. During processing of SWIFT 10K messages the internal buffer can increase above 32KB. Therefore the allocation of an extension area is mandatory in this case. This is simply done by allocating more SPA file records than the required minimum number of records calculated above. The SPA file extension records are shared by all end users.

The size requirement for the SPA file extension depends on the requirements for large message processing in your installation. Assuming that 3 end users are working in parallel with large messages of up to 2MB the total size requirement is 6MB. When the record length is defined as 32700 (default value), the number of the SPA file extension records should be about 193 (6MB divided by 32700). For example, 20 users doing message processing and three of these users working with large messages up to 2MB would result in a space requirement of:

```
20 users * 3 records + 1 index record + 193 extension records  = 254 records.
```

If the space is insufficient, one or more end users processing large messages would be limited to a smaller message size. The maximum number of records supported for the MERVA ESA SPA file is 6474.

The machine-readable material for MERVA ESA contains a sample job for the definition and initialization of the SPA file.

The JCL and the required parameters are described in the *MERVA for ESA Operations Guide*.

Note: You can use your MERVA ESA V3.3 SPA file in MERVA ESA V4.1, provided that the SPA file parameters have not changed.

SPA File in an HDAM Database

MERVA ESA provides a program for using an HDAM database as SPA file. Refer to the chapter on the DSLEUD SPA File Program in the *MERVA for ESA Customization Guide* for details about the structure of this database.

The program DSLEOSPB must be used instead of DSLEOSPA by DSLEUD. To activate DSLEOSPB it must be link-edited with the name DSLEOSPA into a load library in the IMS MPP STEPLIB concatenation.

Calculating the size of the database

Some other characteristics of the database must be defined in the database description (DBD). Refer to the *IMS/ESA Version 5 Administration Guide: Database Manager* for a discussion of these considerations:

- Number of RAPs. Because the number of database records is relatively small, one for each user, it is not necessary to define more than one RAP per block.
- Size of RAA. You should define at least one RAA block for each user. If you specify FRSPC for the data set, you need to define more, but if the number of

users is small, this is unnecessary. The RAA size in the RMNAME parameter includes the bitmap, but not the first VSAM block which is reserved by IMS.

Sample Generation of SPA IMS Database

The example assumes three users and that MERVA ESA has been defined using default values as described in “SPA File (IMS only)” on page 18.

DBD Generation: Note that the DBD specifies two data sets, all SPA5 segments are placed in the second data set. This is because if they exist at all, SPA5 segments might contain large amounts of data compared with the other segments. Placing these segments in a separate data set allows the data set to be tailored specifically for these data volumes.

```
//PROCLIB    JCLLIB ORDER=IMS510.PROCLIB
//STEP1      EXEC DBDGEN,MBR=DSLSPA
//C.SYSLIB   DD DSN=IMS510.MACLIB,DISP=SHR
//C.SYSPRINT DD SYSOUT=*
//C.SYSIN    DD *
         DBD   NAME=DSLSPA,ACCESS=(HDAM,VSAM),                         *
               RMNAME=(DFSHDC40,1,4)        RANDOMISER, RAPS, RAA BLKS
         DATASET DD1=DSLSPA1,DEVICE=3390,SIZE=16384
         SEGM  NAME=SPA1,BYTES=4068,                4090-L'PFX, FIXED   *
               POINTER=TWIN
         FIELD NAME=(LTERM,SEQ,U),BYTES=8,START=17
         SEGM  NAME=SPA2,BYTES=(4084,4),            4090-L'PFX, VAR     *
               PARENT=((SPA1,SNGL)),POINTER=TWIN
         SEGM  NAME=SPA3,BYTES=(4084,4),            4090-L'PFX, VAR TOF *
               PARENT=((SPA1,SNGL)),POINTER=TWIN
         SEGM  NAME=SPA4,BYTES=(4084,4),            4090-L'PFX, VAR LDS *
               PARENT=((SPA1,SNGL)),POINTER=TWIN
         DATASET DD1=DSLSPA2,DEVICE=3390,SIZE=16384
         SEGM  NAME=SPA5,BYTES=(16366,4),           VAR TOF EXT.        *
               PARENT=((SPA1,SNGL)),POINTER=TWIN
         DBDGEN
         FINISH
         END
/*
//L.SYSPRINT DD SYSOUT=*
//L.SYSLMOD  DD DSN=MERVA.DBDLIB(DSLSPA),DISP=SHR
//
```

Figure 2. Sample DBD Generation

Defining the Database Data Sets: Assuming three users must be supported, the VSAM database data sets can be defined in the following way.

Again assuming that the buffers have the default size, three control intervals are sufficient to contain the buffers of one user (excluding the TOF extension buffer for large messages). The first data set contains 11 records, one RAA block and two overflow blocks for each user, the bitmap, and the block reserved by IMS.

The second data set contains sufficient blocks to allow three users to simultaneously process a 2MB message and one bitmap block.

```
//VSAMDEF  EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
 DEL merva_spafile1
 DEL merva_spafile2
 SET MAXCC = 0
 DEFINE  CLUSTER ( -
         VOLUME(vol_id) -
         NAME(merva_spafile1) -
         NONINDEXED -
         RECORDS(11) -
         RECORDSIZE (16377,16377) -
         CONTROLINTERVALSIZE (16384)  )
 DEFINE  CLUSTER ( -
         VOLUME(vol_id) -
         NAME(merva_spafile2) -
         NONINDEXED -
         RECORDS(385) -
         RECORDSIZE (16377,16377) -
         CONTROLINTERVALSIZE (16384)  )
/*
//
```

Figure 3. Cluster Definition for the SPA File Database

In the JCL the lowercase parameters have the following meanings:

merva_spafile1
The data set name of the SPA file database.

merva_spafile2
The data set name of the second SPA file database used for the TOF extension buffer for large messages.

vol_id The volume where the SPA file databases are allocated.

Note: Make sure your IMS defines a VSAM subpool size of at least 16384 bytes. This declaration is made with the VSRBF parameter in the applicable DFSVSMxx member of the IMS proclib. For details see the *IMS/ESA Version 5 Installation Volume 2*.

If the VSAM subpool is declared too small, DSLEOSPB terminates with the message “DSL1037 AI IMS error in program DSLEOSPA”. In the job log of the MPP job you find the message “DFS0730I, reason O,DC”, where DC is the VSAM return code.

Load PSB Generation: To load the database a PSB with PROCOPT=L is required. Figure 4 on page 22 contains an example job stream to generate this PSB:

```
//PROCLIB  JCLLIB ORDER=IMS510.PROCLIB
//STEP1      EXEC PROC=PSBGEN,MBR=DSLSPAL,SOUT=*
//C.SYSIN  DD *
         PCB     TYPE=DB,DBDNAME=DSLSPA,PROCOPT=LS,KEYLEN=8
         SENSEG  NAME=SPA1,PARENT=0
         SENSEG  NAME=SPA2,PARENT=SPA1
         SENSEG  NAME=SPA3,PARENT=SPA1
         SENSEG  NAME=SPA4,PARENT=SPA1
         SENSEG  NAME=SPA5,PARENT=SPA1
         PSBGEN  LANG=ASSEM,PSBNAME=DSLSPAL
         END
/*
//L.SYSLMOD  DD DSN=MERVA.PSBLIB(DSLSPAL),DISP=SHR
//
```

Figure 4. Sample Load PSB Generation

Update PSB Generation: The HDAM database must be added to the MERVA ESA End-User Driver PSB.

```
//PROCLIB    JCLLIB ORDER=IMS510.PROCLIB
//STEP1      EXEC PSBGEN,MBR=DSLEUD,SOUT=*
//C.SYSLIB   DD DSN=IMS510.MACLIB,DISP=SHR
//C.SYSIN    DD *
         PCB   TYPE=TP,MODIFY=YES
         PCB   TYPE=DB,DBDNAME=DSLCORN,PROCOPT=A,KEYLEN=32
         SENSEG NAME=DSLCORN
         PCB   TYPE=DB,DBDNAME=DWSCOR,PROCOPT=A,KEYLEN=24
         SENSEG NAME=DWSCORDA
         PCB   TYPE=DB,DBDNAME=ENLCOR,PROCOPT=A,KEYLEN=24
         SENSEG NAME=ENLCORDA
         PCB     TYPE=DB,DBDNAME=DSLSPA,PROCOPT=A,KEYLEN=8
         SENSEG  NAME=SPA1,PARENT=0
         SENSEG  NAME=SPA2,PARENT=SPA1
         SENSEG  NAME=SPA3,PARENT=SPA1
         SENSEG  NAME=SPA4,PARENT=SPA1
         SENSEG  NAME=SPA5,PARENT=SPA1
             PSBGEN PSBNAME=DSLEUD,LANG=ASSEM,CMPAT=YES
             END
/*
//L.SYSLMOD  DD DSN=MERVA.PSBLIB(DSLEUD),DISP=SHR
//
```

Figure 5. Sample PSB Generation for DSLEUD

ACB Generation: For online access an ACB must be generated. Figure 6 contains an example:

```
//PROCLIB    JCLLIB ORDER=IMS510.PROCLIB
//STEP1      EXEC ACBGEN,SOUT=*
//G.IMS      DD DSN=MERVA.DBDLIB,DISP=SHR
//           DD DSN=MERVA.PSBLIB,DISP=SHR
//G.IMSACB   DD DSN=MERVA.ACBLIB,DISP=SHR
//G.SYSIN    DD *
  BUILD DBD=DSLSPA
  BUILD PSB=DSLEUD
//
```

Figure 6. Sample ACB Generation

Initial Load of the Database: IMS requires a database to be initially loaded before it can be accessed in update mode (PROCOPT=A).

After defining the data sets the database can be loaded using the IMS DFSDDLT0 utility. The following example loads a single root segment with a key of blanks.

Since DSLEOSPB deletes all segments from the database the first time it is called following MERVA ESA startup, this segment is deleted automatically.

```
//STEP1    EXEC PGM=DFSRRC00,PARM='DLI,DFSDDLT0,DSLSPAL',REGION=2M
//STEPLIB  DD  DSN=IMS510.RESLIB,DISP=SHR
//IMS      DD  DSN=MERVA.PSBLIB,DISP=SHR
//         DD  DSN=MERVA.DBDLIB,DISP=SHR
//DSLSPA1  DD  DSN=merva_spafile1,DISP=SHR
//DSLSPA2  DD  DSN=merva_spafile2,DISP=SHR
//IEFRDER  DD  DUMMY
//DFSVSAMP DD *
VSRBF=16384,3
/*
//PRINTDD  DD  SYSOUT=*
//****SYSUDUMP DD  SYSOUT=*
//SYSIN    DD  *
S 1 1 1 1 1    DSLSPA
EH
L       ISRT  SPA1
L       DATA
/*
//
```

Figure 7. Initial Load of the SPA File Database

Large Message Cluster

Definition of the Large Message Cluster - Usage of Free Space

Figure 8 shows sample JCL for defining the LMC with respect to good usage of disk space at the cost of I/O performance.

```
//DEFCSD   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
         DEL lmc           -
         DEF CL (  NAME(lmc) -
                   IXD -
                   CYL(400 0) -
                   SPEED      -
                   reuse      -
                   RECSZ( avgrlen maxrlen) -
                   CISZ(ci_size)       -
                   SHR( 2 ) -
           DATA (  NAME(lmc.DATA) -
                   KEYS( 8 0 )  ) -
          INDEX (  NAME(lmc.INDEX))
/*
```

Figure 8. Sample JCL for LMC Definition - Good Space Usage

In the JCL the lowercase parameters have the following meanings:

lmc The data set name of the LMC cluster.

reuse Required if the old LMC is to be reused after a reorganization process with MERVA ESA being terminated, LMC reorganized, and MERVA ESA restarted.

ci_size

The control interval size of the LMC cluster that can range from 4096 up to 32768 bytes. The recommended value for **ci_size** is *32768*.

avgrlen

Required parameter greater than zero, recommended to be equal to **maxrlen**.

maxrlen

Required parameter determining the segment size of large messages. Messages larger than **maxrlen** are segmented into multiple records with the size of **maxrlen**.

It is recommended to specify **maxrlen** = **ci_size** − 7 to allow VSAM to fully use the space provided by a control interval when storing a message segment.

The recommended value for **maxrlen** is *32761*.

Definition of the Large Message Cluster - Performance

Figure 9 shows sample JCL for defining the LMC with respect to good I/O performance at the cost of usage of free disk space.

```
//DEFCSD   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
         DEL lmc            -
         DEF CL (  NAME(lmc) -
                   IXD -
                   CYL(400 0) -
                   spanned      -
                   SPEED      -
                   reuse      -
                   RECSZ( avgrlen maxrlen) -
                   CISZ(ci_size)        -
                   SHR( 2 ) -
           DATA (  NAME(lmc.DATA) -
                   KEYS( 8 0 )  ) -
          INDEX (  NAME(lmc.INDEX))
/*
```

Figure 9. Sample JCL for LMC Definition - Good I/O Performance

In the JCL the lowercase parameters have the following meanings:

lmc The data set name of the LMC cluster.

spanned

Required, if the **maxrlen** value specified is greater than the specified **ci_size** value. **maxrlen** > **ci_size** is recommended for improving I/O performance. The number of segments per message and thereby the number of I/O operations per message is reduced.

reuse Required, if the old LMC is to be reused after a reorganization process with MERVA ESA being terminated, LMC reorganized, and MERVA ESA restarted.

ci_size

The control interval size of the LMC cluster that can range from 4096 up to 32768 bytes. The recommended value for **ci_size** is *4096*.

avgrlen

Required parameter greater than zero, recommended to be equal to **maxrlen**.

maxrlen

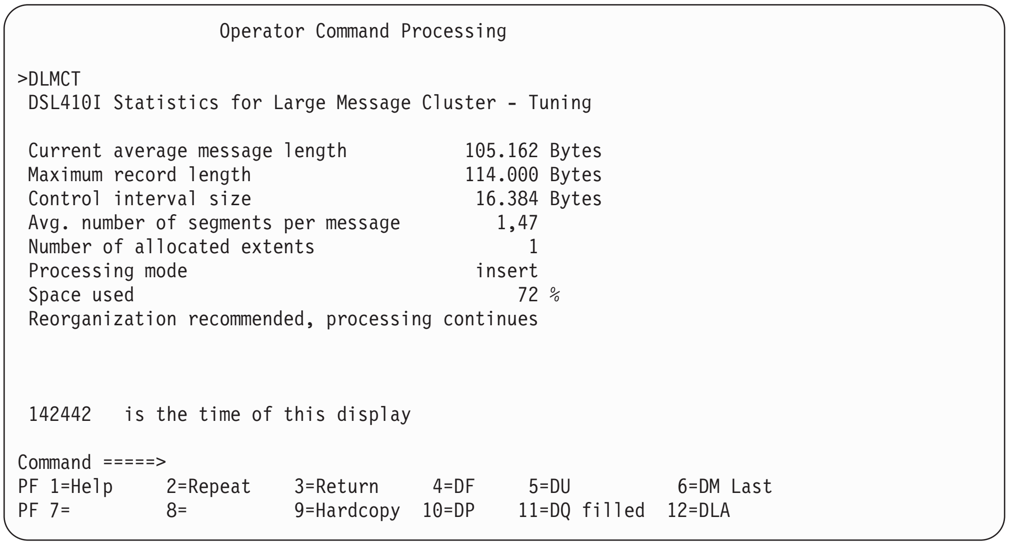
Required parameter determining the segment size of large messages. Messages larger than **maxrlen** are segmented into multiple records with the size of **maxrlen**.

Together with the parameter **spanned** the maximum possible value for **maxrlen** can be equal to approximately the size of the maximum control area for the DASD type in use.

> **Warning**
>
> The LMC full condition can occur along with very poor usage of free space. In the LMC defined with **spanned** only control area splits are done leaving a high probability of failing splits that cause the LMC full condition. An unnecessary big value for **maxrlen** should be avoided. The recommended value for **maxrlen** is the current average message length. The current average message length can be obtained from information displayed when entering the *DLMCT* command.

Example of the Display from a DLMCT Command

Figure 10 shows the information displayed when entering the *DLMCT* command.



```
                   Operator Command Processing

>DLMCT
 DSL410I Statistics for Large Message Cluster - Tuning

 Current average message length           105.162 Bytes
 Maximum record length                    114.000 Bytes
 Control interval size                     16.384 Bytes
 Avg. number of segments per message         1,47
 Number of allocated extents                    1
 Processing mode                           insert
 Space used                                    72 %
 Reorganization recommended, processing continues



 142442   is the time of this display

Command =====>
PF 1=Help     2=Repeat    3=Return     4=DF     5=DU          6=DM Last
PF 7=          8=         9=Hardcopy  10=DP    11=DQ filled  12=DLA
```

Figure 10. Displaying the Status of the LMC for Tuning

Performance Considerations when Processing Large Messages

The *DLMCT* command can be used to get information related to the definition of the large message cluster. The following sections describe how to interpret the information displayed by the *DLMCT* command. The general guidelines can be followed when looking at performance and effectiveness of space usage.

Performance is improved by reducing segmentation and number of I/O operations at the cost of more main and disk storage.

Maximum record length: Defining the maximum possible record size, which is approximately the size of the maximum control area for the DASD in use, will reduce segmentation to a minimum. Messages up to the length of the control area will be written to the LMC with a single I/O in the fastest way. Only messages larger than the control area will be segmented.

Control interval size: Defining the maximum possible control interval size of 32KB will reduce the number of I/O operations to a minimum.

Avg. number of segments per large message: A ratio greater than 1 indicates segmentation of large messages. The aim is to have as little segmentation as possible, which means a ratio close to 1.

Number of allocated extents: A number greater than 1 indicates the possibility of secondary extent allocation allowed for the LMC. For good performance only primary allocation should be allowed by specifying 0 for secondary allocation in the LMC cluster definition.

Processing mode and **Space used**: These fields indicate the following:

- **Processing mode** = load

  No performance degradation independent of the filling status of the LMC.
- **Processing mode** = insert and **Space used** < 70%

  No performance degradation due to the filling status of the LMC.
- **Processing mode** = insert and **Space used** > 70%

  Performance degradation is likely due to the filling status of the LMC. In this case, **Reorganization recommended** is shown in the command response.

Note: It is recommended to allocate a new LMC for MERVA ESA V4.1

MERVA ESA Counter Log Data Set

Although the format of the counter log data set has not changed from MERVA ESA V3.3, it is recommended that you allocate a new counter log data set for MERVA ESA V4.1 and migrate the contents of the MERVA ESA V3.3 data set to the new data set. The migration deletes obsolete records that are older than 12 months. These steps are part of the standard installation procedure.

Installing MERVA ESA with Queue Management Using DB2

Before running MERVA ESA with queue management using DB2 you must install the necessary DB2 objects (tables, views, indices, ..) and establish a connection between your CICS or IMS and DB2. This section covers:

1. MVS installation
   a. Parameters
   b. Step-by-step installation
   c. Sample files and JCL
2. VSE installation
   a. Parameters
   b. Step-by-step installation
   c. Sample files and JCL
3. Table sizes
4. Problem determination

5. Database reorganization and statistics

Parameters you may need to modify are summarized in two tables. The tables show, for MVS and VSE:

- The parameters you need to consider
- Where the parameters must be specified
- The default values that are supplied in the distributed material

The following abbreviation is used in these tables:

RCT Resource control table

Installing the DB2 Objects and the DB2 Connection under MVS

Under MVS you need to consider the following parameters:

Table 1. Parameters for DB2 MVS Objects and Connection

	Parameter	Used In	Default Value	Your Value
1	Buffer pool	DSLDDL02, 04 – 09: BUFFERPOOL BPnn **Note:** Using different values for tables and indices might improve performance.	BP2, BP3	
2	Database	DSLDDL01 – 10: DATABASE cccccccc	DSLDB01 (recommended)	
3	DB2 subsystem name	DSLBIND (bind job): DSN SYSTEM(cccc) DSLPRM: DB2SS=cccc IMS BMP startup job: PROC ...,SSM=cccc EXEC PGM=DFSRRC00,PARM=(...,&SSM) RCT: SUBID=cccc	DB2H	
4	Plan name	DSLBIND job: BIND PLAN(cccccccc) DSLPRM: DB2PLB=cccccccc	DSLNUCxx **Note:** xx is the RCT suffix	
5	Plan owner	DSLBIND (bind job): OWNER(cccccccc)		
6	RCT suffix	RCT: SUFFIX=xx	xx	
7	SQL authorization ID	DSLDDL01, DSLDDL02: SQLID **Note:** Must have SYSADM or SYSCTRL rights	VGDSL	
8	Storage group	DSLDDL01 – 09: STOGROUP cccccccc	DSLSG01 (recommended)	
9	Table qualifier	DSLBIND (bind job): QUALIFIER(cccccccc) DSLDDL04 – 11: SQLID **Note:** This determines the owner of the DB2 tables	 USER	

Step-by-Step Installation

1. Define the DB2 objects (tables, indices, etc.).

   Run DSLDDL02 ·· DSLDDL11 in SPUFI. Adapt the values listed in Table 1 and the size specifications to your needs. For production `WITH RESTRICT ON DROP` is recommended. DSLDDL01 can be used to drop objects. A list of all delivered SPUFI files can be found in “SPUFI Files” on page 29. The DB2 tables are described in detail in the appendix of the *MERVA for ESA Concepts and Components*. For space requirements see “Size Considerations” on page 32.

2. Bind the DB2 application plan.

   An application plan is the control structure used by DB2 to process SQL statements imbedded in application programs. A sample job DSLBIND can be found in the SDSLINS0 library. It is also shown in “DB2 MVS Bind Job” on page 29.

   Note: Under IMS the plan name and the PSB name must be the same (DSLNUCxx).
3. If the user who starts MERVA is different from the owner of the application plan DSLNUCxx (specified with the `OWNER(cccccccc)` parameter in the BIND job), you must grant execute authority for that plan to that user or to PUBLIC, as shown here:

   ```
   GRANT EXECUTE ON PLAN plan-name TO uid
   ```

   where **uid** is either the user ID that is to be given authority, or PUBLIC.
4. Verify that the DB2 CICS language interface module DSNCLI is included in your CSD definitions.
5. Under CICS, define a CICS RCT entry.

   The RCT controls the CICS DB2 interface, in particular you assign a DB2 application plan to a transaction name. For more information on the CICS RCT refer to:

   - Under CICS/ESA: the *CICS/ESA System Definition Guide* and the *DB2 Administration Guide*
   - Under CICS TS V1.1: the *CICS TS System Definition Guide* and the *DB2 Administration Guide*
   - Under CICS TS V1.2 or higher: the *CICS TS Resource Definition Guide* and the *CICS TS DB2 Guide*

   A sample RCT entry is shown in “DB2 MVS RCT Entry” on page 30. Assemble and link-edit it into a STEPLIB library (if you are running CICS/ESA or CICS TS up to V1.1) or a DFHRPL library (if you are running CICS TS V1.2). Starting with CICS TS V1.2 you can also define the CICS DB2 interface using RDO, with CICS TS V1.3 or higher you have to.
6. Change your DSLPRM:

   DB2SS DB2 subsystem name, for example DB2H

   DB2PLB DB2 plan name, for example DSLNUCxx

   QIO DSLQMDIO (MERVA ESA queue management DB2 module)

   Assemble and link-edit it into the MERVA ESA load library. For details on DSLPRM see the *MERVA for ESA Macro Reference*.
7. Under CICS with releases up to CICS TS V1.1, the STEPLIB in the startup job JCL must contain:

   - The CICS SDFHAUTH library
   - The library with the CICS RCT
   - The DB2 load library

   See the program directory for MERVA ESA for details.
8. Under CICS with releases of CICS TS V1.2 or higher:

- The DFHRPL in the startup job JCL must contain the library with the CICS RCT (if not defined using RDO)
- The STEPLIB in the startup job JCL must contain:
  - The CICS SDFHAUTH library
  - The DB2 load library

See the program directory for MERVA ESA for details.

9. Under IMS, the JCL in the BMP job must contain the DB2 load library in the DFSESL and in the STEPLIB concatenation. In the DFSESL concatenation, the DB2 load library must be specified *after* the IMS RESLIB.

   The JCL in the BMP must also identify the DB2 subsystem member (SSM) in the IMS.PROCLIB library with the SSM EXEC parameter.

SPUFI Files

The following files are supplied in the SDSLSAM0 library to allow the DB2 objects to be defined from the TSO SPUFI panel:

DSLDDL01 Drop all DB2 objects

DSLDDL02 Create DB2 storage group DSLSG01 and DB2 database DSLDB01

DSLDDL03 Create DB2 tablespaces DSLTS01 ·· DSLTS06

DSLDDL04 Create DB2 table DSLTQUEL and indices for DSLTQUEL

DSLDDL05 Create DB2 table DSLTQXDEF and indices for DSLTQXDEF

DSLDDL06 Create DB2 table DSLTQXKEY and indices for DSLTQXKEY

DSLDDL07 Create DB2 table DSLTQBUSY and indices for DSLTQBUSY

DSLDDL08 Create DB2 table DSLTQMSG and indices for DSLTQMSG

DSLDDL09 Create DB2 table DSLTQFUN and indices for DSLTQFUN

DSLDDL10 Create DB2 table DSLTQSTAT

DSLDDL11 Create DB2 views

DB2 MVS Bind Job

The following figure shows the MVS JCL to bind the DB2 application plan.

```
//.......  JOB  ............
//*
//*  BIND PLAN DSLNUCxx
//*
//BIND     EXEC PGM=IKJEFT01
//STEPLIB  DD DSN=SYS1.DSNDB2H.LOADLIB,DISP=SHR
//DBRMLIB  DD DSN=MERVA.SDSLDBR0,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
 DSN SYSTEM(DB2H)
 BIND PLAN(DSLNUCxx)    -
 OWNER(cccccccc)        -
 QUALIFIER(cccccccc)    -
 MEMBER(DSLQMDIO,       -
        DSLQMDLI,       -
        DSLQMDXK)       -
 ACQUIRE(USE)           -
 ACTION(REPLACE) RETAIN -
 ISOLATION(CS)          -
 RELEASE(COMMIT)        -
 VALIDATE(BIND)
 END
/*
//
```

Figure 11. Bind DB2 Application Plan Sample JCL (MVS)

DB2 MVS RCT Entry

The following figure shows a sample for a CICS RCT:

```
***********************************************************************
* NAME: DSN2CTXX
***********************************************************************
         DSNCRCT TYPE=INIT,                                            +
               SIGNID=uid,                                             +
               SNAP=A,                                                 +
               STRTWT=NO,                                              +
               SUBID=DB2H,                                      parm3  +
               SUFFIX=xx                                        parm6
         DSNCRCT TYPE=ENTRY,                                           +
               AUTH=(authid),                                          +
               PLAN=DSLNUCxx,                                   parm4  +
               THRDA=1,                                                +
               THRDM=1,                                                +
               THRDS=1,                                                +
               TXID=(DSLN)
         DSNCRCT TYPE=FINAL
         END
```

Figure 12. CICS RCT Sample

Installing the DB2 Objects and the DB2 Connection under VSE

Under VSE you need to consider the following parameters:

Table 2. Parameters for DB2 VSE Objects and Connection

	Parameter	Used In	Default Value	Your Value
1	Database		DSLDB01	
2	DBSPACE	DSLDDVnn	DSLDSnn	
3	Password	DSLDDVnn: CONNECT ···	SQLDBAPW	
4	SQL authorization ID	DSLDDVnn: CONNECT ···	SQLDBA	

Step-by-Step Installation

1. Define the DB2 objects (tables, indices, etc.).

   Run DSLDDV03 ··· DSLDDV11 in ISQL or with the DBS utility. Adapt the values listed in Table 2 and the size specifications to your needs. DSLDDV01 can be used to drop objects. A list of all delivered DBS utility files can be found in "Input Files for the DB2 DBS Utility". For space requirements refer to "Size Considerations" on page 32.
2. Reload the access modules DSLQMDIO, DSLQMDLI, and DSLQMDXK to DB2. A sample job can be found on the installation JCL.

   Depending on the values chosen during installation of your DB2 system, you may have to increase the value of the NCSCANS parameter in your DB2 startup job, for example:

   ```
   // EXEC  ARISQLDS,SIZE=AUTO,PARM='NCSCANS=50'
   ```
3. If the user who starts MERVA is different from the owner of the DB2 packages DSLQMDxx, you must grant execute authority for the corresponding package names to that user or to PUBLIC, as shown here:

   ```
   GRANT EXECUTE ON owner.DSLQMDIO TO uid
   GRANT EXECUTE ON owner.DSLQMDLI TO uid
   GRANT EXECUTE ON owner.DSLQMDXK TO uid
   ```

   where **uid** is either the user ID that is to be given authority, or PUBLIC.
4. Change in your DSLPRM:

   QIO DSLQMDIO

   Assemble and link-edit it into the MERVA ESA load library. For details see the *MERVA for ESA Macro Reference*.
5. Add the DB2 library to the LIBDEF search chain of your CICS startup JCL.

Input Files for the DB2 DBS Utility

The following input files for the DB2 DBS utility are in the MERVA ESA sublibrary:

DSLDDV01	Drop all DB2 objects
DSLDDV02	Dummy, describes all created objects
DSLDDV03	Create DB2 DBSPACE DSLDS01 ·· DSLDS06
DSLDDV04	Create DB2 table DSLTQUEL and indices for DSLTQUEL
DSLDDV05	Create DB2 table DSLTQXDEF and indices for DSLTQXDEF
DSLDDV06	Create DB2 table DSLTQXKEY and indices for DSLTQXKEY

DSLDDV07 Create DB2 table DSLTQBUSY and indices for DSLTQBUSY

DSLDDV08 Create DB2 table DSLTQMSG and indices for DSLTQMSG

DSLDDV09 Create DB2 table DSLTQFUN and indices for DSLTQFUN

DSLDDV10 Create DB2 table DSLTQSTAT

DSLDDV11 Create DB2 views

Size Considerations

The layout of the DB2 tables is described in the appendix of *MERVA for ESA Concepts and Components*. When you define the DB2 tablespaces, you need to consider the following sizes:

DSLTQUEL Each message needs one row in the DSLTQUEL table. Each row is 108 bytes long. There are 4 indices.

DSLTQXDEF Each defined extra-key needs one row in the DSLTQXDEF table. Each row is 60 bytes long plus up to 202 bytes for the CREATECOMMENT VARCHAR column. There is 1 index.

DSLTQXKEY Each extra-key needs one row in the DSLTQXKEY table. Each row is 46 bytes long plus up to 50 bytes for the KEYVALUE VARCHAR column. There are 4 indices.

DSLTQBUSY Each message *in-service* (‘BUSY’) needs one row in the DSLTQBUSY table. Each row is 31 bytes long. There is 1 index.

DSLTQMSG Each message segment needs one row in the DSLTQMSG table. Each row is 11 bytes long plus up to 4002 bytes for the MESSAGE VARCHAR column. There is 1 index.

DSLTQFUN and DSLTQSTAT
Each function needs one row in the DSLTQFUN table. Each row is 18 bytes long. There is 1 index on DSLTQFUN. DSLTQSTAT has only one row with 26 bytes.

Problem Determination

1. DB2 attachment
   - Under MVS CICS, you must start the CICS DB2 attachment before you start MERVA ESA.

     If the message

     ```
     DSL006A MERVA startup failed in program 8, RC is 172
     ```

     is displayed, start the DB2 attachment

     ```
     DSNC STRT xx
     ```

     where xx is the suffix specified with the SUFFIX parameter in the CICS RCT.

     If the message DSL006A is displayed and is preceded by the message

     ```
     DSL492E DSLSRVP cmd=LOAD, mod=DSNCLI, rc=X'0004', rs=X'004D'
     ```

     verify that the DB2 CICS language interface module DSNCLI is included in your CICS CSD definition.
   - Under VSE CICS, you must start the CICS DB2 attachment before you start MERVA ESA.

If MERVA startup fails with CICS abend code AEY9, start the DB2 attachment: `CIRB`.

- Under IMS, there is no need to start the DB2 attachment. You can check the status of the DB2 attachment with the command `/DIS SUBSYSTEM subsystem-name`.

2. If message ‘DSL006A MERVA startup failed in program 8, RC is 168’ is displayed, check the journal for SQL errors DSL491E. Maybe a (re-) bind is necessary.
3. DQ SQLERROR command

   Details of the last SQL error can be displayed with the MERVA **dq sqlerror** command. See the *MERVA for ESA Operations Guide* for details.
4. DQ STATUS command

   The status of the MERVA ESA queue database can be displayed with the MERVA **dq status** command. See the *MERVA for ESA Operations Guide* for details.

DB2 Database Statistics

For best performance you should reorganize the database from time to time, and then run the DB2 RUNSTATS utility (MVS) or issue the `UPDATE STATISTICS` statement (VSE), and rebind the DSLNUCxx plan.

Installing the DSLSDIR and DSLSDOR ISPF Panels (MVS)

Like DSLSDI, DSLSDIR reads a batch of messages from a sequential data set and writes them to MERVA ESA queues.

Like DSLSDO, DSLSDOR reads a batch of messages from a MERVA ESA queue and writes them to a sequential data set.

DSLSDIR and DSLSDOR are described in the *MERVA for ESA Operations Guide*. The utilities can be invoked via JCL and, under MVS, from ISPF panels. To invoke them from ISPF panels, you have to first install a few files on MVS data sets.

DSLSDIR comprises the following files:

No.	Name	Type	Description
1	DSLSDIR	REXX exec	The main exec
2	DSLSDIRF	REXX exec	Exec to display front-end panel
5	DSLSDIRC	ISPF panel	Pop-up window to enter jobcards
3	DSLSDIRD	ISPF panel	Pop-up window to enter DD statements
4	DSLSDIRH	ISPF panel	Help panel
6	DSLSDIRP	ISPF panel	Front-end panel

DSLSDOR comprises the following files:

No.	Name	Type	Description
1	DSLSDOR	REXX exec	The main exec
2	DSLSDORF	REXX exec	Exec to display front-end panel
5	DSLSDORC	ISPF panel	Pop-up window to enter jobcards
3	DSLSDORD	ISPF panel	Pop-up window to enter DD statements

No.	Name	Type	Description
4	DSLSDORH	ISPF panel	Help panel
6	DSLSDORP	ISPF panel	Front-end panel

Installation Steps

1. Before you install the DSLSDIR and DSLSDOR ISPF panels, you have to decide how you want the front-end execs DSLSDIRF and DSLSDORF to locate the ISPF panels:
   - By looking in a data set allocated to **ISPPLIB**. This is a little faster at execution time, but requires more installation effort.
   - By calling the ISPF service **LIBDEF** at execution time. This is a little slower at execution time, but saves effort during installation.

   Note: To list all currently allocated DD-names, use the TSO command LISTALC STATUS.
2. If you prefer the **ISPPLIB** method, do the following:
   a. Copy the front-end execs DSLSDIRF and DSLSDORF to a data set allocated to SYSPROC or SYSEXEC.
   b. Copy the ISPF panels DSLSDIRC, DSLSDIRD, DSLSDIRH, DSLSDIRP, DSLSDORC, DSLSDORD, DSLSDORH, and DSLSDORP to a data set allocated to ISPPLIB.

   If you prefer the **LIBDEF** method, do the following:
   a. Copy the front-end execs DSLSDIRF and DSLSDORF to a data set allocated to SYSPROC or SYSEXEC.
   b. Copy the ISPF panels DSLSDIRC, DSLSDIRD, DSLSDIRH, DSLSDIRP, DSLSDORC, DSLSDORD, DSLSDORH, and DSLSDORP to any fixed blocked, record length 80 (FB80) dataset, or leave them in the SDSLSAM0 data set.
   c. Edit DSLSDIRF and DSLSDORF. Go to the Customization Section and set:
      1) The variable **libdef** to YES
      2) The variable **panellib** to the name of the data set to which you copied the ISPF panels in step 2

      For example:

```
libdef   = 'YES'                        /* LIBDEF 'NO' or 'YES'     */
panellib = 'UID.MERVA.PANELS'           /* MERVA ISPF panel library */
```

3. You can now invoke the DSLSDIR and DSLSDOR ISPF panels from ISPF option 6 with the commands **%DSLSDIRF** and **%DSLSDORF**, respectively.
4. If you have the REXX/370 Compiler (5695-013) and Library (5695-014) installed, recompiling DSLSDIR and DSLSDOR will improve their performance.

Chapter 4. Installing the SWIFT Link

You can ignore this chapter if you do not want to use the SWIFT Link.

SWIFT Link together with MERVA ESA is a complete pregenerated system that, after installation, can be run and tested. Some of the SWIFT Link tables have to be adapted to your installation environments. Refer to “Chapter 10. Generating MERVA ESA Tables” on page 61 to find out how to make the necessary changes.

To implement the system according to the needs of your organization and operating environment, refer to the *MERVA for ESA Customization Guide*, the *MERVA for ESA Operations Guide*, and the *MERVA for ESA Macro Reference*.

Calculating the Space for the SWIFT Link Data Sets

This section shows methods for calculating the space for allocating the SWIFT Link data sets.

Authenticator-Key File

The authenticator-key file is optional. If the parameter AUTHENT=NO is specified in the DWSPARM macro in the SWIFT Link customizing parameter module DWSPRM, the authentication services of the SWIFT Link are not used, and the authenticator-key file is not necessary.

The VSAM cluster definition requires one VSAM record for each authenticator record you want to store. This depends on:

- How many banks with their own authenticator keys share the SWIFT Link.
- With how many correspondents each of these banks communicates.
- How many group correspondents are used.

Note: If you want to insert additional authenticator records later, you must allow for sufficient free space in the VSAM control intervals. It is recommended to multiply the calculated number of records by two. If eventually VSAM is no longer able to insert new records because all control intervals are in use, reorganize the cluster using the IDCAMS REPRO function.

The machine-readable material for the SWIFT Link contains the JCL for defining the VSAM cluster and formatting the authenticator-key file. You can run the job again with a different number of records if you want to adjust the capacity of the authenticator-key file.

To load the authenticator-key file:

- Use the SWIFT Bilateral Key Exchange (BKE) functions of MERVA ESA Components running under Windows NT®, sending the authenticator keys from MERVA ESA Components to MERVA ESA.
- Use the authenticator-key file online maintenance functions described in *MERVA for ESA User's Guide*.
- Create a sequential input file with ADD input records for DWSAUTLD, which is described in the *MERVA for ESA Operations Guide*.
- Load the authenticator keys from the MERVA ESA V3.3 authenticator-key file:

1. Unload the file to a sequential data set using the MERVA ESA V3.3 DWSAUTLD program, which is described in the *MERVA ESA V3.3 Operations Guide*.
2. Reload the sequential file to the authenticator-key file using the MERVA ESA V4.1 SWIFT Link DWSAUTLD program, which is described in the *MERVA for ESA Operations Guide*. Make sure that the same unload password is used for both unload and reload (defined with the AUTHUP parameter of the DWSPARM macro in the customizing parameter module DWSPRM).

Note: You must format the authenticator-key file using DWSAUTLD before you can use it. If BKE is used, you must format it with the correct secure transmission key (STK).

SWIFT Correspondents File

The SWIFT Correspondents File is optional. This file need only be defined and filled if address expansion is defined in the MERVA ESA function table. The record length of this file differs in CICS and IMS:

- In CICS, the record length is 1738 bytes.
- In IMS, the record length is 1746 bytes according to the RECORDSIZE for the DBD generation.

The SWIFT Correspondents File is loaded from either the SWIFT BIC Directory Update tape or the SWIFT BIC Database Plus tape. The number of records in the Correspondents File is determined by the number of unique BIC codes contained on the tape. Currently there are about 30000 BIC codes on the tape, and this number will increase in the future. Therefore, the SWIFT Correspondents File should be allocated with 30000 to 35000 records.

The machine-readable material for the SWIFT Link contains the JCL for defining and initializing of the SWIFT Correspondents File. Loading the SWIFT Correspondents File from a SWIFT BIC tape is described in the *MERVA for ESA Operations Guide*.

Note: You can use your MERVA ESA V3.3 SWIFT Correspondents File with MERVA ESA V4.1.

SWIFT Currency Code File

The SWIFT Currency Code File is optional. This file need only be defined and filled if currency code checking using the file is defined in the MERVA ESA parameter table.

The record length of this file differs in CICS and IMS:

- In CICS, the record length is 1622 bytes.
- In IMS, the record length is 1630 bytes according to the RECORDSIZE for the DBD generation.

The SWIFT Currency Code File is loaded from either the SWIFT BIC Directory Update tape or the SWIFT BIC Database Plus tape. The number of records in the Currency Code File is determined by the number of currency records contained on the tape. Currently there are about 200 such records on the tape.

The machine-readable material for the SWIFT Link contains the JCL for defining and initializing of the SWIFT Currency Code File. Loading the SWIFT Currency Code File from a SWIFT BIC tape is described in the *MERVA for ESA Operations Guide*.

Note: You can use your MERVA ESA V3.3 SWIFT Currency Code File with MERVA ESA V4.1.

Chapter 5. Installing the Telex Link

If you do not want to use the Telex Link, ignore this chapter.

Telex Link uses the Telex Correspondents File, an optional data set which is used for the address expansion when entering data into the telex header area.

The machine-readable material for the Telex Link contains the JCL for the VSAM cluster definition and the initialization of the Telex Correspondents File.

To define the data set, the number of senders and correspondents known in the installation has to be determined. This number is used as the number of records in the VSAM cluster definition of the Telex Correspondents File. Afterwards the file must be filled with correspondents data. This can be done using the MERVA ESA file maintenance function FLM.

Note: You can use your MERVA ESA V3.3 Telex Correspondents File with MERVA ESA V4.1.

In MERVA ESA there are two ways to communicate with the public telex network:

- The Telex Link via workstation
- The Telex Link via a fault-tolerant system using the HeadOffice Telex

Information about installing Telex Link via workstation can be found in *MERVA Workstation Based Functions*.

The rest of this chapter deals with Telex Link via a fault-tolerant system only, which allows MERVA ESA to communicate with the public telex network via a fault tolerant front-end computer. The Telex Link material is contained in the MERVA ESA distribution material.

Telex Link is a complete pregenerated system that, after successful installation, can be run and tested. Some of the Telex Link tables and additions to MERVA ESA tables have to be adapted to your installation environment. Refer to "Chapter 10. Generating MERVA ESA Tables" on page 61 to find out how to make the necessary changes and how the MERVA ESA table generation process works.

To implement the system according to the needs of your organization and operating environment, refer also to the *MERVA for ESA Customization Guide*, the *MERVA for ESA Operations Guide*, and the *MERVA for ESA Macro Reference*.

The Telex Link and the Telex Substation

The Telex Link does not communicate directly with the public telex network. It exchanges messages with a Telex Substation. Usually this substation is an unattended, continuously operational gateway to the public telex network.

ACF/VTAM Definitions

The line speed of the connection to the Telex Substation must be 4800 or 9600 bps.

Figure 13 shows a sample VTAM® definition of the line to the Telex Substation for CICS, and the MODEENT definition in the MODETAB.

```
************************************************************************
* DEFINITION OF THE TELEX SUBSTATION
************************************************************************
*
I4210037 LINE  ADDRESS=(037,HALF)    via fault tolerant front-end
         SERVICE ORDER=(I4S222)
I4S222   PU    PUTYPE=2,ADDR=C1,MAXDATA=265,MAXOUT=7,                  *
               PASSLIM=7,ISTATUS=ACTIVE,
*
I4S32201 LU    PACING=1,VPACING=1,LOCADDR=1
------------------------------------------------------------------------
* MODEENT DEFINITION USED FOR Telex Link FOR CICS
* DEFINED AS A 3278 MODEL 2 SCREEN
D4C32782 MODEENT LOGMODE=D4C32782,FMPROF=X'03',TSPROF=X'03',           *
               PRIPROT=X'B1',SECPROT=X'90',COMPROT=X'3080',            *
               RUSIZES=X'8585',PSERVIC=X'024000000000185000007E00'
```

Figure 13. Sample Line Definition

Modify the netname (I4S32201) as required by your installation.

Figure 14 shows a sample VTAM definition of the line to the Telex Substation under IMS, and the MODEENT definition in the MODETAB.

```
I4210036 LINE  ADDRESS=(036,HALF)
         SERVICE ORDER=(I4S221)
I4S221   PU    PUTYPE=2,ADDR=C1,MAXDATA=265,MAXOUT=7,DLOGMOD=IMSS1,    *
               PASSLIM=7,ISTATUS=ACTIVE,MODETAB=YYYYYYY
*
I4S32101 LU    PACING=1,VPACING=1,LOCADDR=1

---------------------------------------------------------------------
* MODEENT DEFINITION USED FOR Telex Link FOR IMS
* SLUTYPE P
IMSS1    MODEENT LOGMODE=IMSS1,FMPROF=X'04',TSPROF=X'04',              *
               PRIPROT=X'B1',SECPROT=X'B1',COMPROT=X'6080',            *
               PSERVIC=X'000000000000000000000000'
```

Figure 14. Sample Line Definition for NCP and MODEENT Definition

Modify the MODETAB name (YYYYYYY) as required by your installation.

Note: In IMS a character translation from %% to CRLF (X'0D25') during Telex receive processing is performed. This is done to avoid IMS translation of CRLF and requires adequate setup in the telex substation program.

Chapter 6. Installing MERVA Link ESA and the FMT/ESA Using MERVA Link

Installing MERVA Link ESA

The material for the MERVA Link functions executing in the CICS and IMS environments (MERVA Link ESA) is contained in the MERVA ESA distribution material.

MERVA Link ESA together with MERVA ESA is a complete pregenerated system. After successful installation as described in the *Program Directory for MERVA ESA*, and the necessary customization of CICS or IMS, you can run the MERVA Link ESA installation verification test described in “Chapter 14. Verifying the Installation of MERVA Link” on page 79.

The material for the MERVA Link functions executing in the OS/390 UNIX System Services environment (MERVA Link USS) is also contained in the MERVA ESA distribution material.

However, MERVA Link USS must be installed in a separate installation task. The installation of MERVA Link USS is described in “Chapter 9. Installing MERVA Link USS” on page 53. After the MERVA Link USS installation you can run the MERVA Link USS installation verification test described in “Chapter 18. Verifying the Installation of MERVA Link USS” on page 107.

Note: When the term **MERVA Link** is used in this book without the qualifier **USS**, **MERVA Link ESA** is meant.

Installing the FMT/ESA Using MERVA Link

The FMT/ESA material is contained in the MERVA ESA distribution material.

Before you can use the FMT/ESA it is a prerequisite that MERVA Link is installed. Moreover, it is recommended that the MERVA Link installation has been successfully verified. FMT/ESA together with MERVA ESA is a complete pregenerated system. After successful installation as described in the *Program Directory for MERVA ESA*, you can run the installation verification test described in “Chapter 15. Verifying the Installation of FMT/ESA Using MERVA Link” on page 85.

Chapter 7. CICS Requirements

Main Storage Requirements

For MERVA ESA:

- Some MERVA ESA programs are loaded below 16MB and are defined with EXECKEY(CICS) in the CICS program definitions. MERVA ESA needs about 3MB in this area.
- Some MERVA ESA programs allocate main storage which is required to reside below 16MB. 1MB is usually sufficient for MERVA ESA.
- Most of the storage allocated by MERVA ESA transactions is taken from above 16MB. When large messages are to be processed, this area has to be defined large enough. The minimum requirement for MERVA ESA is three times the value specified in the MAXBUF parameter of the DSLPRM customization module. In addition, for each active task processing large messages up to two times the value specified in the MAXBUF parameter is needed. A minimum of 4MB should be specified for MERVA ESA.
- Reentrant tables, MCBs, and some programs of MERVA ESA are loaded above 16MB. A minimum of 4MB should be reserved for MERVA ESA.

For MERVA ESA under CICS/ESA 4.1 and Later Releases

The relevant parameters are DSALIM and EDSALIM. MERVA ESA runs in a CICS region with a minimum of 7MB below the 16MB line and a minimum of 20MB above the 16MB line.

If the SWIFT Link is used, the first line to the SWIFT network needs about 500KB, each additional line about 300KB above 16MB outside of the CICS controlled storage (EDSALIM). Make sure that the required amount of storage is available to MERVA ESA SWIFT Link.

For MERVA ESA under VSE

The CICS partition must have enough GETVIS storage available for the XPCC interregion communication. This can be calculated using the NICPL and MAXBUF parameters of the DSLPARM macro and the ECB parameter of the interregion entry in the nucleus program table (TYPE=INTER). Required storage is:

```
72  +  ( (NICPL + MAXBUF + 280) × ECB )
```

When MERVA ESA applications run as CICS transactions in a different partition from the MERVA ESA nucleus, that partition must also have enough GETVIS storage available for the XPCC interregion communication. The calculation depends on the number of requests that can be issued at the same time. (The maximum number of requests is equal to the number of interregion ECBs.) Required storage is:

```
(NICPL + MAXBUF + 396) × requests
```

If the SWIFT Link with X.25 for the connection to the SWIFT network is used, the GETVIS storage must be increased by 500KB for the first line to the SWIFT network and by 300KB for each additional line. Each SNA session to MERVA Extended Connectivity running on a 37xx controller represents one line.

CICS Temporary Storage

The MERVA ESA end-user transaction saves its working storage into the CICS temporary storage data set between conversation steps. Each signed-on end user needs about 64KB plus the actual size of the message in process. For example, consider an installation with 12 end users working in parallel:

- 10 end users working with messages up to 32KB in size
- 2 end users working with messages up to 2MB in size.

The space (in KB) needed in the temporary storage data set is:

```
10 × ( 64 + 32 ) + 2 × ( 64 + 2048 ) = 960 + 4224  = 5184
```

The MERVA ESA intertask communication method using CICS temporary storage queues also uses the CICS temporary storage. The maximum size for an active session during communication can be calculated by adding the value of NICPL and MAXBUF of the DSLPRM customization parameter module; the recommended minimum value is 250KB.

Normally it is not needed to reserve this space for all possible sessions. The maximum number of active sessions is determined by the ECB parameter in the DSLNPT definition of the CICSSRV nucleus server.

The storage is only required during the short period where a MERVA ESA intertask communication takes place. In most cases the actual buffers are smaller than the maximum value. The available storage can be shared by several tasks. If this method for intertask communication is used, the minimum value to be added to the CICS temporary storage is 250KB.

CICS Definitions

The CICS definitions required by MERVA ESA and its components are supplied with the machine-readable material in the following form:

DSLSNT	Signon table sample for CICS
DSLSNT41	Signon table sample for CICS/ESA 4.1
xxxPCT21	Transaction definitions for PCT (VSE only)
xxxPPT21	Program definitions for PPT (VSE only)
xxxFCT21	File definitions for FCT (VSE only)
xxxTCT21	Terminal definitions for TCT (VSE only)
xxxDCT21	Destination definitions for DCT (VSE only)
xxxCSD23	CICS/VSE 2.3 CSD utility definition statements
xxxCSDTS	CICS TS 1.1 for VSE CSD utility definition statements
xxxCSD41	CICS/ESA 4.1 CSD utility definition statements

The following naming conventions apply:

- The member names starting with DSL are for Base Functions
- The member names starting with DWS are for the SWIFT Link
- The member names starting with ENL are for the Telex Link
- The member names starting with EKA are for the MERVA Link and FMT/ESA using MERVA Link

Member names with a 2-digit number at the end are CICS release-dependent. For example, the number 41 indicates CICS Version 4 Release 1. MERVA ESA under CICS/ESA provides appropriate definition statements for the CICS/ESA CSD utility program DFHCSDUP. There are three different sets of CSD definitions. The batch job to install these definitions is described in the *Program Directory for MERVA ESA* provided with the distribution material.

When running this job, the following groups for MERVA ESA are defined in the CSD:

DSLGROUP For the Base Functions definitions

DWSGROUP For the SWIFT Link definitions

ENLGROUP For the Telex Link definitions

EKAGROUP For the definitions for MERVA Link and FMT/ESA using MERVA Link

Append the groups required for your installation to the LIST used for the CICS/ESA startup job (GRPLIST parameter of the DFHSIT macro). For details refer to *CICS/ESA Version 4.1 Resource Definition (Online)*.

For MERVA ESA under CICS/ESA 4.1 and later releases

The CICS signon table is no longer available with CICS/ESA 4.1. The information stored in the DFHSNT of earlier releases of CICS can be stored in the CICS segment of the RACF database for CICS/ESA 4.1. CICS/ESA 4.1 provides a migration tool to migrate a DFHSNT of previous CICS versions to the CICS segment in the RACF database. Refer to the *CICS/ESA CICS-RACF Version 4.1 Security Guide* for further information.

MERVA ESA uses the OPIDENT parameter of the DFHSNT to verify the user's authorization to start MERVA ESA. This feature can be switched off by specifying CID=*** in the DSLPRM customizing parameter module. If this feature should be used, the appropriate information must be migrated into the RACF database, as described above.

The following table shows the CICS tables to be changed in your installation for MERVA ESA and its components:

CICS Table	Member Name	Comments
Signon Table (SNT)	DSLSNT DSLSNT41	The OPIDENT parameter must be the same as the CID parameter in the macro DSLPARM of the MERVA ESA customizing parameter module DSLPRM.
System Initialization Table (SIT)		Increase the MXT parameter by the number of ECBs generated in DSLNPTT (see the MNOTE in the assembly listing of DSLNPTT). As MERVA ESA uses dynamic transaction backout, the DBUFS2 parameter of DFHSIT must be at least 2048 to avoid performance degradation.

CICS Table	Member Name	Comments
Program Control Table (PCT)	DSLPCT21 DWSPCT21 ENLPCT21 EKAPCT21	These copy codes contain the necessary PCT entries for the Base Functions (DSLPCT21), SWIFT Link (DWSPCT21), Telex Link (ENLPCT21), and MERVA Link (EKAPCT21).
Processing Program Table (PPT)	DSLPPT21 DWSPPT21 ENLPPT21 EKAPPT21	These copy codes contain the necessary PPT entries for the Base Functions (DSLPPT21), SWIFT Link (DWSPPT21), Telex Link (ENLPPT21), and MERVA Link and FMT/ESA using MERVA Link (EKAPPT21). They also contain entries for the online help MCBs for user and operator messages. If this online help service is not required, remove these PPT entries. If you change a module name in DSLPRM, DWSPRM, or ENLPRM, you must change the PPT accordingly.
File Control Table (FCT)	DSLFCT21 DWSFCT21 ENLFCT21	These copy codes contain the necessary FCT entries for the MERVA ESA nicknames file (DSLFCT21), the SWIFT Correspondents File (DWSFCT21), and the Telex Correspondents File (ENLFCT21). For each file you want to process using the MERVA ESA general file services also include an entry in the FCT.
Destination Control Table (DCT)	EKADCT21	These copy codes contain the sample definitions for the conversation trace TD queue.
Terminal Control Table (TCT)	ENLTCT21 EKATCT21	For MERVA ESA running under VSE, the VSE system console must be defined as a terminal in the TCT, if the startup and operator commands of MERVA ESA are processed from the VSE console. For a correct adjustment of the console messages, LPLEN=80 must be specified. When using the hardcopy function, check the printer definitions in the MERVA ESA Terminal Feature Definition Table (DSLTFDT). See the *MERVA for ESA Customization Guide* for details. The ENLTCT21 copy codes contain the TCT entry for the line between the Telex Link and Headoffice Telex on a fault-tolerant system. In particular, the definition of the transaction code ENLR for the received telex messages is necessary. The transaction code ENLR is supplied by Headoffice Telex on a fault-tolerant system when sending a message to the Telex Link. EKATCT21 contains sample definitions for the MERVA Link connections. These sample definitions are for reference purposes only. Do not include these TCT entries in an active TCT if these entries are not modified.

The following table describes how to use Resource Definition (Online) (RDO), when installing MERVA ESA:

CICS RDO	Member Name	Comments
For MVS and CICS/ESA 4.1 or subsequent release	DSLCSD41 DWSCSD41 EKACSD41 ENLCSD41	The job to install the required MERVA ESA definitions is part of the MERVA ESA installation described in the program directory for MERVA ESA, which is provided with the distribution material. MERVA ESA programs must be defined with EXECKEY(CICS).
For VSE and CICS/ESA 2.3 or subsequent release	DSLCSD23 DWSCSD23 EKACSD23 ENLCSD23	The job to install the required MERVA ESA definitions is part of the MERVA ESA installation described in the program directory for MERVA ESA, which is provided with the distribution material. MERVA ESA programs must be defined with EXECKEY(CICS).
For VSE and CICS TS 1.1 or subsequent release	DSLCSDTS DWSCSDTS EKACSDTS ENLCSDTS	The job to install the required MERVA ESA definitions is part of the MERVA ESA installation described in the program directory for MERVA ESA, which is provided with the distribution material. MERVA ESA programs must be defined with EXECKEY(CICS).

Additional CICS Requirements for the Telex Link via a Fault-Tolerant System

1. For the CICS transaction definitions, you must consider the following:
   a. You must also specify the name of the Telex Link Send transaction (ENLS) in:
      - The MERVA ESA function table entry for the Telex Link Send queue (TXHCFSND)
      - The customizing parameter module ENLPRM, with the parameter STRAN of the ENLPARM macro

   b. You must also specify the name of the Telex Link Receive transaction (ENLR) in:
      - The CICS Terminal Control Table (TCT) entry for the line to Headoffice Telex on a fault-tolerant system, using the TRANSID parameter
      - The customizing parameter module ENLPRM, using the RTRAN parameter of the ENLPARM macro
      - Headoffice Telex on a fault-tolerant system, for messages sent from the Headoffice Telex on a fault-tolerant system to the Telex Link
2. For the terminal definitions in CICS, you must consider the following:
   - The TRMIDNT parameter for the definition of the line between the Telex Link and Headoffice Telex on a fault-tolerant system must also be used in the LTERM parameter of the DSLFNT macro when defining the Telex Link Send queue TXHCFSND in the MERVA ESA function table.

- In the CICS Terminal Control Table (TCT), you must include the following parameters in the DFHTCT TYPE=INITIAL macro:

  ACCMETH=(...,VTAM)
  VTAM must be included as the access method.

  APPLID=*name*
  name is a 1- to 8-character name that is referred to by Headoffice Telex on a fault-tolerant system, and must therefore also be defined when customizing Headoffice Telex on a fault-tolerant system.

Additional CICS Requirements for MERVA Link

To run the MERVA Link installation verification test as described in "Chapter 14. Verifying the Installation of MERVA Link" on page 79, a set of MERVA Link resources (programs and transactions) must be defined to CICS. These definitions are in copy books EKAPPT21 and EKAPCT21, or in EKACSD41, depending on the CICS version used.

If you want to run the MERVA Link installation and verification test only, you need not include the copy books EKATCT21 or EKADCT21 into the CICS tables DFHTCT or DFHDCT. Refer to the *MERVA for ESA Customization Guide* for a detailed description of the CICS customization for MERVA Link.

Required CICS System Parameter

If you define an interconnection between your CICS system and a partner system using an LU 6.2 intersystem connection, you must specify ISC=YES in the DFHSIT of your CICS system, or as a CICS startup parameter.

Chapter 8. IMS Requirements

The following tables describe the IMS definitions required to run MERVA ESA.

IMS Nucleus Generation

Member Name	Comments
DSLIMSAP	DSLIMSAP contains the necessary APPLCTN and TRANSACT macro instructions for MERVA ESA. The transaction codes DSLE, DSLH, DSLP, DSLX, and DWSA can be changed. The code used instead of DSLE must also be defined in the MERVA ESA customization parameter module DSLPRM. The codes used instead of DSLH, DSLX, and DWSA must also be defined in the MERVA ESA Function table. The MSGTYPE parameter can also be changed. For DSLE, RESPONSE is used (and recommended), but NONRESPONSE can be used instead. DSLIMSAP also contains the necessary APPLCTN and TRANSACT macro instructions for the Telex Link program ENLHCF1. The transaction codes ENLR and ENLS can be changed. The code used instead of ENLR must also be defined in the customizing parameter module ENLPRM and in Headoffice Telex on a fault-tolerant system for messages sent from Headoffice Telex on a fault-tolerant system to the Telex Link. The code used instead of ENLS must also be defined in the Telex Link Send queue, in the MERVA ESA function table, and in the customizing parameter module ENLPRM. The MSGTYPE=(,,16) can also be changed. The first two subparameters of MSGTYPE must be (MULTSEG,NONRESPONSE), which are the defaults. If you install a test-key processing program that requires definitions in IMS, you can include these definitions at the same time. DSLIMSAP also contains the necessary APPLCTN and TRANSACT macro instructions for the MERVA Link programs. The transaction codes used (EKASxx) must also be defined in the MERVA Link send queues defined in the MERVA ESA function table, and in the MERVA Link partner table. DSLIMSAP also contains the necessary APPLCTN and TRANSACT macro instructions for the MERVA-MQI Attachment programs. The code used instead of DSLS must also be defined in the MERVA-MQI Attachment send and start queues defined in the MERVA ESA function table. The code used instead of DSLR must also be defined in the MERVA-MQI Attachment start queues defined in the MERVA ESA function table, and in the MQSeries process definitions.
ENLCOMM	ENLCOMM contains the necessary definitions for the line between the Telex Link and Headoffice Telex on a fault-tolerant system. The line is a nonswitched SDLC line. The terminal name of the line between the Telex Link and Headoffice Telex on a fault-tolerant system must also be defined in the Telex Link Send queue TXHCFSND in the MERVA ESA function table.

Note: When defining screen terminals in the IMS nucleus, the **OUTBUF** parameter of IMS must be 500 greater than the **BUFSIZE** parameter of the DSLTFD macroinstruction.

DBDGEN

Member Name	Comments
DSLEIDBD	Contains the JCL for the DBD generations of: • MERVA ESA Nicknames File • MERVA ESA SWIFT Link Correspondents File • MERVA ESA Telex Correspondents File If you do not want to use one or more of these files in your system, remove the appropriate generation steps from this JCL member.
DSLDBD	If used, generates DBDs for the MERVA ESA Nicknames File.
DWSDBD	If used, generates DBDs for the SWIFT Correspondents File.
DWSDBDCU	If used, generates DBDs for the SWIFT Currency Code File.
ENLDBD	Generates the PCB TYPE=DB statement for the Telex Correspondents File, which is used by the MERVA ESA program DSLEUD.
xxxDBD	Generates a DBD for each additional file using MERVA ESA General File services.

PSBGEN

The PSBs for the programs DSLCXT, DSLFLUT, DSLKQR, DSLKQS, DSLPTMRV, and DWSCORUT are only required when these programs are used. In all PSBs, the PCBs for the MERVA ESA Nicknames File, the SWIFT Correspondents File, the SWIFT Currency Code File, and the Telex Correspondents File are only required in a program's PCB if it uses these files. This depends on the specifications of the EXPAND and EXPNAM parameters of the DSLFNT macroinstruction in the MERVA ESA function table, and, for the PSB of DSLEUD, on the use of these files for MERVA ESA General File maintenance. The SWIFT Currency Code File is used only when the parameter CURCODE=FILE is coded in the MERVA ESA customization module DSLPRM.

Member Name	Comments
DSLEIPSB	Contains the JCL for the PSB generations of all MERVA ESA base programs and databases, as well as those for SWIFT Link, Telex Link, and MERVA Link. If you do not want to use one or more of these components, remove the appropriate generation step(s) and/or COPY statements from this JCL member.
DSLPSBCX DWSPSBCX	Generates the PSB for the checking and expansion transaction DSLCXT. DSLPSBCX contains the PCB TYPE=TP statement. DWSPSBCX contains the PCB TYPE=DB statement for the SWIFT Correspondents File and for the SWIFT Currency Code File.
xxxPSBCX	A PCB must be generated for each file used with DSLCXT.
DSLPSBEU DWSPSBEU	Generates the PSB for the MERVA ESA End-User Driver transaction DSLEUD. DSLPSBEU contains the PCB TYPE=TP statement and the PCB TYPE=DB statement for the MERVA ESA Nicknames File. If this file should not be used, remove the PCB TYPE=DB statement for DSLCORN. DWSPSBEU contains the PCB TYPE=DB statement for the SWIFT Correspondents File and for the SWIFT Currency Code File.
ENLPSBEU	Contains the PCB TYPE=DB statement for the Telex Correspondents File. PCBs for databases needed for the test-key processing program must be included in the PSB of DSLEUD after the PCB of the Telex Correspondents File. The position of the first of these PCBs must be specified as the TKPPCB parameter in the customizing parameters ENLPRM.

Member Name	Comments
xxxPSBEU	A PCB must be generated for each file used with DSLEUD.
DSLPSBFG DWSPSBFG	Generate the PSB for the MERVA ESA General File utility DSLFLUT used for listing files. DSLPSBFG contains the PCB TYPE=DB, PROCOPT=G statement for listing the MERVA ESA Nicknames File. DWSPSBFG contains the PCB TYPE=DB, PROCOPT=G statement for listing SWIFT Correspondents File and the SWIFT Currency Code File.
ENLPSBFG	Contains the PCB TYPE=DB, PROCOPT=G statement for listing the Telex Correspondents File.
xxxPSBFG	A PCB must be generated for each file used with DSLFLUT.
DSLPSBFL DWSPSBFL	Generates the PSB for the MERVA ESA General File utility DSLFLUT used for initializing files. DSLPSBFG contains the PCB TYPE=DB, PROCOPT=L statement for initializing the MERVA ESA Nicknames File. DWSPSBFG contains the PCB TYPE=DB, PROCOPT=L statement for initializing SWIFT Correspondents File and the SWIFT Currency Code File.
ENLPSBFL	Contains the PCB TYPE=DB, PROCOPT=L statement for initializing the Telex Correspondents File.
xxxPSBFL	A PCB must be generated for each file used with DSLFLUT.
DSLPSBHC DWSPSBHC	Generate the PSB for the MERVA ESA hardcopy print transaction DSLHCP. DSLPSBHC contains the PCB TYPE=TP statement. DWSPSBHC contains the PCB TYPE=DB statement for the SWIFT Correspondents File and for the SWIFT Currency Code File.
xxxPSBHC	A PCB must be generated for each file used with DSLHCP.
DSLPSBNU DWSPSBNU	Generates the PSB for the nucleus program DSLNUC. DSLPSBNU contains the PCB TYPE=TP statement. DWSPSBNU contains the PCB TYPE=DB statement for the SWIFT Currency Code File.
DWSPSBAT	Generates the PSB for the Authenticator-Key File Update Transaction program DWSAUTT.
DSLPSBES DSLPSBSE	Generates the PSBs for the SWIFT to EDIFACT conversion Transaction programs DSLCESTR and DSLCSETR.
DSLPSBPT	Generates the PSB for the program-to-MERVA switch transaction DSLPTMRV.
DSLPSBKR DWSPSBKR	Generates the PSB for the MERVA-MQI Attachment receive transaction DSLKQR. DWSPSBKR contains the PCB TYPE=DB statement for the SWIFT Currency Code file.
DSLPSBKS DWSPSBKS	Generates the PSB for the MERVA-MQI Attachment send transaction DSLKQS. DWSPSBKS contains the PCB TYPE=DB statement for the SWIFT Currency Code file.

ACBGEN

Member Name	Comments
DSLEIACB	Contains the JCL for the ACB generations of all MERVA ESA base programs and databases, as well as those for SWIFT Link, Telex Link, and MERVA Link. If you do not want to use one or more of these components, remove the appropriate BUILD statements from this JCL member.
None	For each file you want to process with the MERVA ESA General File services, an entry in the ACB is required. For the MERVA ESA Nicknames File use: BUILD DBD DSLCORN For the SWIFT Correspondents File use: BUILD DBD DWSCOR For the SWIFT Currency Code File use: BUILD DBD DWSCUR For the Telex Correspondents File use: BUILD DBD ENLCOR

Database Generation

Member Name	Comments
DSLIMSDB	Contains the necessary DATABASE macroinstructions for the Nicknames File, the SWIFT Currency Code File, the SWIFT and Telex Correspondents Files to be used in the IMS system generation. Include this copy book into your database definitions.
DSLIMSDA	You can request dynamic allocation in the IMS dynamic allocation member for each file you want to process with the MERVA ESA General File services. DSLIMSDA contains the necessary IMS macroinstructions for the MERVA ESA Nicknames File, the SWIFT Correspondents File, the SWIFT Currency Code File, and the Telex Correspondents File. Include this copy book into your IMS dynamic allocation definitions.

Chapter 9. Installing MERVA Link USS

This chapter contains information related to the installation of the MERVA ESA functions executing in the OS/390 UNIX System Services (USS) environment. This information covers:

- Installing the MERVA Link USS functions based on an OS/390 partitioned data set (PDS) unloaded from the MERVA ESA machine readable material (MRM) during the MERVA ESA installation.
- Creating a MERVA USS instance.

Installing MERVA Link USS in the HFS

MERVA Link USS can be installed in any directory of the OS/390 USS Hierachical File System (HFS). This directory is called the MERVA USS installation directory.

You can choose either a system-wide directory (for example, **/usr/lpp/merva**) or a subdirectory of your home directory (for example, **$HOME/lpp/merva**) as the MERVA USS installation directory. You may prefer to use the latter alternative because root user authority is required to modify the **/usr** file system. The sample MERVA USS installation directory is **/usr/lpp/merva**.

The MERVA USS installation directory contains static MERVA USS resources, such as MERVA Link USS programs, libraries, and samples. Dynamic MERVA USS resources are associated with a MERVA USS instance. They are contained in a MERVA USS instance directory, which is described later in this chapter.

The MERVA USS installation directory contains a README file that provides last-minute information and hints regarding installation.

The subdirectories of the MERVA USS installation directory are:

bin containing MERVA USS directly callable programs

lib containing MERVA USS dynamic link libraries

cmd containing MERVA USS shell scripts and related files

The procedure to install the static MERVA USS resources consists of several steps.

Unload MERVA USS Materials from the MERVA ESA MRM

As part of the MERVA ESA installation, a PDS is unloaded from the MERVA ESA distribution tape that contains the MERVA USS distribution materials. The name of this OS/390 MVS data set is hlq**.SDSLUSS0**. The sample MERVA ESA high-level qualifier (hlq) is **MERVA**.

Create the MERVA USS Installation Directory

If it is not already available, the MERVA USS installation directory must be created. The sample USS shell commands for this step read:

```
cd /usr/lpp
mkdir merva
```

OS/390 USS root user authority is required to create the sample MERVA USS installation directory in the **/usr/lpp** directory. Root user authority may not be required if you use another directory as MERVA USS installation directory.

Set the MERVA USS Installation Environment Variable

MERVA USS functions use an OS/390 USS environment variable at various places to identify the MERVA USS installation directory. The name of the variable is **DSLPP_DIR**.

The USS shell command to set this variable to the sample MERVA USS installation directory reads:

```
export DSLPP_DIR=/usr/lpp/merva
```

You may wish to add this command to your **.profile** or **.setup** files in order to facilitate working with MERVA USS.

Expand the Tape Archive

The MERVA USS tape archive file must be expanded into the MERVA USS installation directory. There are two ways to do this:

- From an HFS file
- Directly from a MERVA ESA data set (possible only on OS/390 V2R8 and later releases)

Note: OS/390 USS root user authority may be required to perform the following steps.

From an HFS file

First, copy the OS/390 MVS PDS member **hlq.SDSLUSS0(DSLTARZ)** to an OS/390 HFS file, for example, **/tmp/merva.tarz**. This HFS file contains a compressed USS tape archive. The sample USS shell command for this step is:

```
tso -t "oput 'merva.sdsluss0(dsltarz)' '/tmp/merva.tarz' binary"
```

Next, expand the tape archive. The USS shell commands for this step are:

```
cd $DSLPP_DIR
rm -fr *
pax -r -p am -f /tmp/merva.tarz
```

The command **rm -fr *** removes all files and subdirectories in the MERVA USS installation directory. It can be omitted if this directory is empty.

From a MERVA ESA Data Set

On OS/390 V2R8 and later releases, you can expand the tape archive directly from a MERVA ESA data set. This obviates the need to copy the tape archive to an HFS file.

The USS shell commands for this step are:

```
cd $DSLPP_DIR
rm -fr *
pax -r -p am -f "//'merva.sdsluss0(dsltarz)'"
```

The command **rm -fr *** removes all files and subdirectories in the MERVA USS installation directory. It can be omitted if this directory is empty.

Define Symbolic Link to the Installation Directory

To minimize the typing effort when working with the MERVA USS installation directory, you may wish to define a symbolic link to this directory in the HFS root directory. The sample OS/390 USS command to define this symbolic link reads:

```
ln -s $DSLPP_DIR /merva
```

If this symbolic link is defined, you can always refer to the MERVA USS installation directory by specifying the symbolic link **/merva**. The shell scripts provided in the **/cmd** subdirectory of the MERVA USS installation directory do not use this symbolic link.

OS/390 USS root user authority is required to define a symbolic link in the HFS root directory. As an alternative that does not require root user authority, you may wish to define this symbolic link in your home directory:

```
ln -s $DSLPP_DIR $HOME/merva
```

Read the README File

Refer to the README file in the MERVA USS installation directory for last-minute information and instructions concerning the installed MERVA USS product.

Installing MERVA Link USS in a Program Object Library

The MERVA Link programs installed in an HFS directory can execute in an OS/390 USS and in an OS/390 BPXBATCH environment. As an alternative, MERVA Link USS programs can execute in a native OS/390 MVS environment. However, the executables in the **/bin** and **/lib** subdirectories of the MERVA USS installation directory must be copied to a Program Object Library in an OS/390 Partitioned Data Set Extended (PDSE) before they can be executed in a native OS/390 MVS environment.

Installing MERVA Link USS in a PDSE is an optional task that should be performed only if there is a need to execute MERVA Link USS programs in the OS/390 MVS environment.

For more information about copying executables from the HFS to a PDSE, and about executing MERVA Link USS programs in the OS/390 MVS environment, refer to *MERVA for ESA Advanced MERVA Link*.

Creating a MERVA USS Instance

When MERVA USS has been installed, a MERVA USS instance must be created. A MERVA USS instance is defined by both of the following:

- MERVA USS Instance Owner, a standard OS/390 USS user
- MERVA USS instance directory, a subdirectory of the user’s home directory or the user’s home directory itself

Creating a MERVA USS instance and working with it requires that the name of the MERVA USS installation directory is available from the OS/390 USS environment variable **DSLPP_DIR**. The following procedure assumes that this variable is correctly set.

Create USS User and the USS Shell Environment

The following tasks must be executed only if the corresponding resources are not already available.

Define and Logon as MERVA USS Instance Owner

A MERVA USS instance is owned by a standard OS/390 USS user who is called the MERVA USS Instance Owner. The sample MERVA USS instance owner is **merva1**.

Define the MERVA USS instance owner or select an existing USS user, and open a USS shell for this user. From now on, you are the MERVA USS instance owner.

Select and Make the MERVA USS Instance Directory

You can choose either your home directory (for example, **/u/merva1**), a subdirectory of your home directory (for example, **/u/merva1/merva1**), or any other directory you have write access to as the MERVA USS instance directory. You may prefer to use the second alternative because it adds only one file to your home directory, and the added files can be deleted with a single USS command. The MERVA USS instance owner's home directory (**/u/merva1**) is the sample MERVA USS instance directory.

Set the MERVA USS Instance Environment Variable

MERVA USS functions use an OS/390 USS environment variable to identify the MERVA USS instance. The name of the variable is **MERVA_DIR**. It must contain the name of the MERVA USS instance directory. The sample instance directory is **/u/merva1**.

The sample USS shell command to set this variable reads:

```
export MERVA_DIR=/u/merva1
```

You may wish to add this command to your **.profile** or **.setup** files in order to facilitate working with MERVA USS.

Define MERVA USS Command Aliases

A shell command script is provided in the **/cmd** subdirectory of the MERVA USS installation directory that defines a set of USS shell command aliases. These command aliases simplify the commands to be entered in the following steps of the MERVA USS instance generation, and the MERVA USS installation verification commands.

The USS shell commands to define the MERVA USS command aliases read:

```
cd $DSLPP_DIR/cmd
. ekadca.cmd
```

You must specify a period and a blank before the command script name **ekadca.cmd** to execute the script in your shell. Otherwise, the aliases are defined in a child process and you cannot use these definitions in your shell.

You may wish to add these two commands to your **.profile** or **.setup** files in order to define these command aliases permanently in all your USS shells.

Create MERVA USS Instance Subdirectories

A shell command script is provided in the **/cmd** subdirectory of the MERVA USS installation directory to create all mandatory and optional subdirectories of the MERVA USS instance directory. Existing subdirectories are re-used. Existing files that may already be customized remain unchanged.

The MERVA USS instance subdirectories must be created by the MERVA USS instance owner. The short form of the USS shell command to create the MERVA USS instance subdirectories reads:

```
csd
```

The full form of this command must be used if the MERVA USS command aliases have not been defined. It reads:

```
$DSLPP_DIR/cmd/ekacsd.cmd
```

The following subdirectories are available from the MERVA USS instance directory when this command has been successfully executed:

cfg configuration files

cmd command scripts

err error logs

ipc inter-process communication resources

sec security files

trc trace and report files

No data is written to the **/cfg**, **/err**, **/sec**, and **/trc** subdirectories. IPC resource files are generated in the **/ipc** subdirectory, and three files are copied to the **/cmd** subdirectory (if not already available).

IPC Resource Files

IPC resource files are created in the **/ipc** subdirectory of the MERVA USS instance directory. The file names are **ekaact.a**, **ekaact.b**, and **ekaact.v**. IPC resource files are empty. They are not supposed to be modified nor to contain data.

Command Files

Command files are copied from the installation directory to the **/cmd** subdirectory of the MERVA USS instance directory if they are not already available in this subdirectory. The file names are **ekatpi.cmd**, **ekatpi.env**, and **ekatci.cmd**. Command files in the instance directory are supposed to be customized by the MERVA USS instance owner to reflect the environment of the MERVA USS instance.

The command files **ekatpi.cmd** and **ekatci.cmd** can be used without modification to start the inbound MERVA Link USS messaging servers for SNA APPC and TCP/IP respectively, if the sample MERVA USS directory names are used. The MERVA USS instance and installation directories must be specified as command parameters if other directory names apply. You may, however, wish to customize the command files and specify the MERVA USS instance and installation directories explicitly.

The copied environment file **ekatpi.env** contains the sample MERVA USS instance directory name. It cannot be used without modification if you don’t use the sample instance directory name.

Basic TCP/IP Customization for MERVA Link USS

MERVA Link USS uses the services of the Internet Daemon (InetD) provided by OS/390 USS to start an inbound TP. TCP/IP and InetD must be customized to support MERVA Link USS. OS/390 USS root user authority is required to perform this customization.

Define TCP/IP Client Network Service

The MERVA Link USS Messaging Server must be customized as a TCP/IP client network service in **/etc/services**. Use a text editor to add the following line to this file:

```
ekamsg1     7110/tcp         # MERVA Link USS Messaging Service
```

Configure InetD Subserver

The MERVA Link USS TCP/IP Messaging Server must be configured as an OS/390 USS InetD subserver in **/etc/inetd.conf**. The InetD subserver configuration entry for this MERVA Link server must specify both the MERVA USS installation and instance directories. There are several alternatives for the subserver configuration. Two alternatives are:

Alternative 1

Call the program **ekatci** directly from the installation directory. The sample InetD subserver configuration entry is:

```
ekamsg1 stream tcp nowait merva1 /usr/lpp/merva/bin/ekatci ekatci /u/merva1
```

Alternative 2

Call the command script **ekatci.cmd** unmodified from the instance directory. The sample InetD subserver configuration entry is:

```
ekamsg1 stream tcp nowait merva1 /bin/sh ekatci /u/merva1/cmd/ekatci.cmd
```

Notes:

1. The parameters **stream**, **tcp**, and **nowait** define TCP/IP socket characteristics.
2. The fourth parameter (**merva1** in the sample) determines the subserver process user. Replace this parameter with the ID of a user who has access to the MERVA USS resources; for example, the MERVA USS instance owner.
3. For Alternative 2, the default parameters of **ekatci.cmd** are the sample names of the MERVA USS instance and installation directories **/u/merva1** and **/usr/lpp/merva**, respectively. If you don't use the sample names, you can specify the actual names as command parameters in the InetD subserver configuration entry, or modify the ekatci command script in the MERVA USS instance directory. Do not modify any resource in the MERVA USS installation directory, as your modification may be overwritten by a MERVA USS maintenance process.
4. Alternative 2 cannot be used by systems that use OS/390 UNIX-Level Security. With such systems, the MERVA Link USS inbound TCP/IP TP (program ekatci) must be called directly from the installation library, as in Alternative 1, not indirectly via /bin/sh and a command script, as in Alternative 2. This is because a command script cannot be program-controlled, and cannot provide the clean address space that ekatci needs. For more information, see "Systems Using OS/390 UNIX-Level Security" on page 59.

Refresh InetD Process

The Internet Daemon (InetD) must be informed of an update of the InetD configuration to activate the update immediately. This is done by sending a signal (SIGHUP) to the InetD process.

1. To find the InetD process_id, enter:

   ```
   ps -e | grep inetd
   ```

2. To send a SIGHUP to the InetD process, enter:

   ```
   kill -1 process_id
   ```

   where *process_id* is the ID returned from the previous **ps** command. The parameter **-1** of the kill commands indicates that the signal is a SIGHUP.

Systems Using OS/390 UNIX-Level Security

Installations with strict security requirements typically use *OS/390 UNIX-Level security*. MERVA Link USS is affected by such systems because the MERVA Link USS InetD subserver uses the the following functions:

__passwd()
To verify a remote user's authorization to access MERVA Link services

setuid()
To change the security environment of the MERVA Link receiving process

To see whether your installation uses OS/390 UNIX-Level security, enter the command

```
RL FACILITY BPX.DAEMON ALL
```

This command returns information that indicates whether BPX.DAEMON is defined in the RACF FACILITY class, and whether the UNIX® kernel (OMVSKERN or OMVSGRP) has READ access to this resource. If so, your system uses OS/390 UNIX-Level security.

MERVA Link USS Program-Controlled Executables

The MERVA Link USS inbound TCP/IP program **ekatci** checks whether the user ID and password of the remote user is valid in the local system. It can perform this check only if it executes in a clean address space. An InetD subserver is scheduled by InetD in a clean address space if InetD executes in a clean address space and if the InetD subserver is program-controlled.

The InetD subserver address space can execute functions that require a controlled environment as long as no uncontrolled program or DLL is loaded into the address space.

The program **ekatci** calls functions from DLL **ekacsc** (MERVA Link USS Common Subroutine Collection) before it executes security related functions. This is why the two MERVA Link USS executables **ekatci** and **ekacsc** must be defined as being program-controlled. The sample USS commands to be issued for this purpose are:

```
extattr +p /usr/lpp/merva/bin/ekatci
extattr +p /usr/lpp/merva/lib/ekacsc
```

Note: These commands can only be issued by a user with authority to set the program-control extended attribute in HFS executables. To grant a user this authority, issue the following commands:

```
REDEF FACILITY BPX.FILEATTR.PROGCTL UACC(NONE)
PERMIT BPX.FILEATTR.PROGCTL CLASS(FACILITY) ID(xxx) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

where **xxx** is replaced with the user ID of the user to whom this authority is to be granted.

The program-control flags must be set again when maintenance has been applied to the MERVA Link USS installation library.

To display the extended attributes of the MERVA Link USS executables, issue the following commands:

```
ls -E /usr/lpp/merva/bin/*
ls -E /usr/lpp/merva/lib/*
```

The program attributes displayed for program ekatci read, for example:

```
-rwxr-x--- -ps
```

The second group of attributes shows the extended attributes. The p-flag identifies an executable as program-controlled. The s-flag (program runs in a shared address space) is set by default when the executable is made by the OS/390 UNIX **make** facility.

The program-controlled attribute need not be set for any other MERVA Link USS executable than **ekatci** and **ekacsc**.

Intentional Destruction of the Program-Controlled Environment

A MERVA Link USS receiving TCP/IP process must execute in a program-controlled environment until all security-related functions have been executed. However, after this is done, etkatci intentionally destroys the program-controlled environment so that security controls are reinstated. It does this by calling the program **ekap1i**, which is not program-controlled, and therefore causes the address space to become dirty. After the boundary between ekatci and ekap1i has been passed once, a MERVA Link USS receiving process can no longer issue security-related functions.

Chapter 10. Generating MERVA ESA Tables

Most of the MERVA ESA environment is defined in tables. When a network link or a user modification is added to MERVA ESA, some or all of these tables are changed.

Some of the tables used in MERVA ESA and its components reflect the organization to be supported by MERVA ESA. Therefore, you may not want to use the sample provided. Instead, you must change some of the tables to suit your organization, or you can code your own tables and change the table name parameters in DSLPRM.

Refer to "Part 4. Maintaining MERVA ESA and Its Components" on page 111 to find out how to make your changes to the system with the help of SMP/E for MVS, or MSHP for VSE.

The MERVA ESA table generation process (DSLGEN) makes it easier to change the following tables:

DSLEPTT	End-User Driver Program and Command Table
DSLEMCMT	Command Table of DSLEMSG
DSLFDTT	Field Definition Table
DSLFLTT	File Table
DSLFNTT	Function Table (sample)
DSLMPTT	MFS Program Table
DSLMSGT	Operator and Error Message Table
DSLMTTT	Message Type Table
DSLNCMT	Nucleus Command Table
DSLNPTT	Nucleus Program Table
DSLNTRT	Task Server Request Table

Note: The MERVA ESA function table is a sample. When a network link is installed, more sample functions are added. When customizing your own function table, you should code it outside the DSLGEN process and give it a name different from DSLFNTT.

Table 3 on page 62 shows which of these tables are link-edited to another module, or if they are loaded by the programs that use them.

Table 3. Table Names and Load Modes

Table Name	Code	Load Module	Link-edited or Loaded
DSLEPTT	EPT	DSLEUD	link-edited
DSLEMCMT	MCM	DSLEUD	link-edited
DSLFDTT	FDT	DSLFDTT	loaded
DSLFLTT	FLT	DSLFLTT	loaded
DSLFNTT	FNT	DSLFNTT	loaded
DSLMPTT	MPT	DSLMMFS	link-edited
DSLMSGT	MSG	DSLMSGT	loaded
DSLMTTT	MTT	DSLMTTT	loaded
DSLNCMT	NCM	DSLNUC	link-edited
DSLNPTT	NPT	DSLNUC	link-edited
DSLNTRT	NTR	DSLNUC	link-edited

If MERVA ESA is installed from the machine-readable material (MRM) tape, all tables will contain the definitions for MERVA ESA Base Functions, SWIFT Link, Telex Link, MERVA Link, and FMT/ESA.

Generating MERVA ESA Tables in VSE

In VSE, generating MERVA ESA tables is a two-stage process. The output from first stage is punched and is the input for the second stage. Each MERVA ESA table that can be modified is built using the table generation macro (DSLGEN).

Stage 1

The first stage generation comprises the following steps:

1. Modify the DSLGEN control statements to suit your installation.

   See “Appendix A. DSLGEN Stage-1 Generation Macro” on page 121 and “Appendix B. MERVA ESA Table Generation Examples” on page 129 for a further explanation including examples of the table generation process.
2. Assemble these statements. The output consists of the following:
   - A combination of statements (INITIAL, COPY, and FINAL) constituting the table or tables you want to build
   - The JCL required to assemble these tables
   - The JCL required to link-edit all load modules

The output of the first stage is a job stream for the second stage.

Stage 2

Submit the set of jobs punched within the first stage process. When the jobs are completed, the modified tables are implemented.

Note: For assembly of MERVA ESA tables or user exits you must specify SYSPARM='CICSVSE' on the OPTION card.

Generating MERVA ESA Tables in MVS

In MVS, generating MERVA ESA tables is a two-stage process. Stage 1 is the table generation, stage 2 the SMP/E installation. Both stages can run as a single job stream, or each stage may run as a separate job. This allows for inspection and modification of the output resulting from stage 1, which is stored in the partitioned data set with last qualifier 'SDSLGEN0'. One or more tables can be installed using this process. See “Appendix A. DSLGEN Stage-1 Generation Macro” on page 121 and “Appendix B. MERVA ESA Table Generation Examples” on page 129 for examples and a further explanation of the table generation process.

Each MERVA ESA table that you can modify is built with the table generation macro (DSLGEN). An overview of the MERVA ESA table generation process follows:

- Ensure that pre-allocation of all required data sets is completed
- Modify the DSLGEN control statements to suit your installation
- Modify the generation JCL to suit your installation
- Submit the generation job stream

The two stages of table generation are:

Stage 1

The first stage generation comprises the following steps:

- DSLGEN statements are assembled. The output consists of a sequential data set containing IEBUPDTE control statements and inline source members. These source members are:
  - SMP/E control statements (member SMPCNTL).

    When a table has already been installed by the DSLGEN process, an SMP/E restore command is generated. SMP/E receive and apply commands are generated for all tables to be installed as a user modification (SMP/E USERMOD). The SYSMOD-ID of a modified table is UDSL*xxx*, where *xxx* is one of the codes shown in the table in Table 3 on page 62.

    The information about the user modification in the SMP/E CSI is deleted immediately after the APPLY. The SYSMOD entries in the global zone and the target zone are deleted and the RMID entry in the SRC and MOD entries of the tables is changed. This is done to allow a subsequent DSLGEN process without having to restore the earlier applied user modification.
  - SMP/E hold input (member SMPHOLD).

    This member consists of ++SRC SMP/E modification control statements for each table modified in this DSLGEN process.
  - SMP/E input (member SMPPTFIN).
  - All modified tables as requested in this DSLGEN process.
- IEBUPDTE stores all source members into a partitioned data set (PDS). The data set name of this PDS has SDSLGEN0 as last qualifier.

  When stage 1 is complete, check that the members in the PDS are correct.

Stage 2

This is the SMP/E installation process. The generated tables are SMP/E user modifications (USERMODs). SMP/E performs all the necessary assembly and link-edit steps for the tables that you have requested in this DSLGEN process.

After installation is complete, a UCLIN command deletes the SYSMOD information.

Note: When not using SMP/E for assembly of MERVA ESA tables or user exits, you must specify the SYSPARM parameter to the assembler program. For programs running under MVS, specify SYSPARM=MVS.

Chapter 11. Customizing MERVA ESA

The final step in the installation prior to validation is customizing MERVA ESA to adapt it to your organization. This is described in the *MERVA for ESA Customization Guide*.

Part 3. Validation of the Installation and Customization

Chapter 12. The MERVA ESA Queue Test and Journal Display Commands

Operators working at a screen terminal (MSC or CMD functions) can use the following commands during installation testing to display and manipulate messages in the MERVA ESA queues:

Queue test commands

COPY Copy messages from one MERVA ESA queue to another.

DELETE Delete messages from a MERVA ESA queue.

DELX Delete messages that start with the specified QSN from a MERVA ESA queue.

FREE Free messages in a MERVA ESA queue, and reset the *in service* indicator.

INPUT This command is no longer supported by MERVA ESA. To fill a queue with test messages, use the batch utilities DSLSDI or DSLSDIR, or a batch program or REXX EXEC that uses MERVA ESA API services.

MOVE Move messages from one MERVA ESA queue to another.

QHELP Display help information about these test commands.

Journal display command

JRN Display a MERVA ESA journal record online, or check the current status of the journal file.

In a production environment:

- Do not use the queue test commands, as they bypass the message flow defined by the routing modules. To limit use of the queue test commands, set the value of the parameter EXQUE in the module DSLPRM to one of:

  MASTER
  Specifies that the queue test commands are allowed for master users only.

  NO Specifies that the queue test commands are not allowed.

  YES Specifies that the queue test commands are allowed for all users.

- Do not use the journal display command, as it lets operators view the contents of the journal. To remove the journal display command from the system, set the value of the parameter EXJRN in the module DSLPRM to NO. When EXJRN=NO is specified, the journal status command **JRN STATUS** is still allowed.

Copying Messages (COPY)

Use the **copy** command to copy messages from one MERVA ESA queue to another. The messages remain in the source queue. It ignores the HOLD or *in service* status of the queues. You can duplicate test messages by specifying the same queue name for the source and the target queue. The copied messages get new UMRs when UMR processing is activated in MERVA ESA.

Command Format

copy **cpy**	*queue1 queue2 number [UMR]*

Parameter Description

queue1
: This is the name of the source queue.

queue2
: This is the name of the target queue.

number
: This is the number of messages to be copied. Copying is either stopped when this number is reached, or when all the messages from the source queue are copied. When the source and target queue names are identical, copying will stop only when the number is reached.

UMR
: This parameter is effective only if MERVA ESA Queue Management is customized for using the Unique Message Reference (UMR). If UMR is specified, all copied messages get a new unique message reference (UMR). If UMR is not specified and a message has no UMR, it will be copied without UMR.

Example

To copy 20 messages from queue DMPR0 to queue DMPR1, enter:

```
copy dmpr0 dmpr1 20
```

The response shows the number of messages copied:

```
DSL1200 Queue Test: 20 messages(s) copied
```

Deleting Messages from the Beginning of a Queue (DELETE)

Use the **delete** command to delete messages from a queue. It ignores the HOLD or *in service* status of the queue.

Command Format

delete **del**	*queue number*

Parameter Description

queue
: This is the name of the queue from which messages are to be deleted.

number
: The number of messages to be deleted. If omitted, 1 is assumed.

Example

To delete 20 messages from queue DMPR1, enter:

```
delete dmpr1 20
```

The response shows the number of messages deleted:

```
DSL1200 Queue Test: 20 messages(s) deleted
```

Deleting Messages Starting from a QSN (DELX)

Use the **delx** command to delete messages from a queue, starting with the specified QSN. It ignores the HOLD or *in service* status of the queue.

Command Format

delx	*queue qsn number*

Parameter Description

queue
This is the name of the queue from which messages are to be deleted.

qsn
This specifies the first QSN to be deleted. The messages are deleted unconditionally, regardless of whether they are invalid or defective.

number
The number of messages to be deleted. If omitted, 1 is assumed.

Example

To delete 20 messages from queue DMPR1 starting with QSN 100, enter:

```
delx dmpr1 100 20
```

The response shows the number of messages deleted:

```
DSL1200 Queue Test: 20 messages(s) deleted
```

Freeing Messages (FREE)

Use the **free** command to reset the *in service* indicator of messages in a queue without altering the messages. The **free** command ignores the HOLD status of the queue.

Command Format

free	*queue number*

Parameter Description

queue
This is the name of the queue in which messages are to be freed.

number
The process of resetting the *in service* indicator is either stopped when this number is reached, or when all messages from the source queue are freed. For example, the command **free** *queue* **9999** usually frees all messages in a queue.

Example

To free 4 messages in queue L1DE0, enter:

```
free l1de0 4
```

The response shows the number of messages freed:

```
DSL1200 Queue Test: 4 messages(s) freed
```

.

Moving Messages (MOVE)

Use the **move** command to move messages from one queue to another while ignoring the HOLD or *in service* status of the queues. As the messages in the target queue will not be in service, you can reset the *in service* indicator for all the messages in a queue by specifying the same name for both the source and target queue.

Command Format

move **mov**	*queue1 queue2 number [UMR]*

Parameter Description

queue1
This is the name of the source queue.

queue2
This is the name of the target queue.

number
This is the number of messages to be moved. Moving is either stopped when this number is reached, or when all messages from the source queue are moved. When the source and target queue names are identical, moving will stop only when the number is reached.

UMR
This parameter is effective only if MERVA ESA Queue Management is customized for using the Unique Message Reference (UMR). If UMR is specified, all moved messages get a new unique message reference (UMR). If UMR is not specified, the UMR of the message will not be changed.

Example

To move 20 messages from queue DMPR0 to queue DMPR1, enter:

```
move dmpr0 dmpr1 20
```

The response shows the number of messages moved:

```
DSL1200 Queue Test: 20 messages(s) moved
```

Inputting Messages (INPUT)

This command is no longer supported by MERVA ESA. To fill a queue with test messages, use the batch utilities DSLSDI or DSLSDIR, or a batch program or REXX EXEC that uses MERVA ESA API services.

Requesting Help Information (QHELP)

The **qhelp** command requests help information about other test commands.

Command Format

qhelp **hlp**	

Parameter Description

The **qhelp** command has no parameters.

Displaying Journal Contents (JRN)

The **jrn** command lets you inspect the MERVA ESA journal during testing of the installation. For example, you can use it display the routing trace or messages sent over the line.

When a journal record is segmented in the MERVA ESA journal data set, this command displays one segment at a time. It places a new **jrn** command in the command line which directly accesses the next journal record or journal record segment. You can press ENTER to sequentially scan the journal file.

Command Format

jrn	[{*datetime*}] [{*FIRST*}] [{*STATUS*}]

Parameter Description

datetime
: This is the date and time value of the journal record that you want displayed. The date format is YYYYMMDD. When a time value is given, it must be concatenated directly. No blanks are allowed between the date and time. The next record with a key higher or equal to the parameter is displayed. When the parameter is omitted, the next journal record is displayed. If no record has been displayed earlier, the first journal record from the current startup of MERVA ESA is displayed.

FIRST
: Enter the keyword **FIRST** to display the first journal record from the current startup of MERVA ESA.

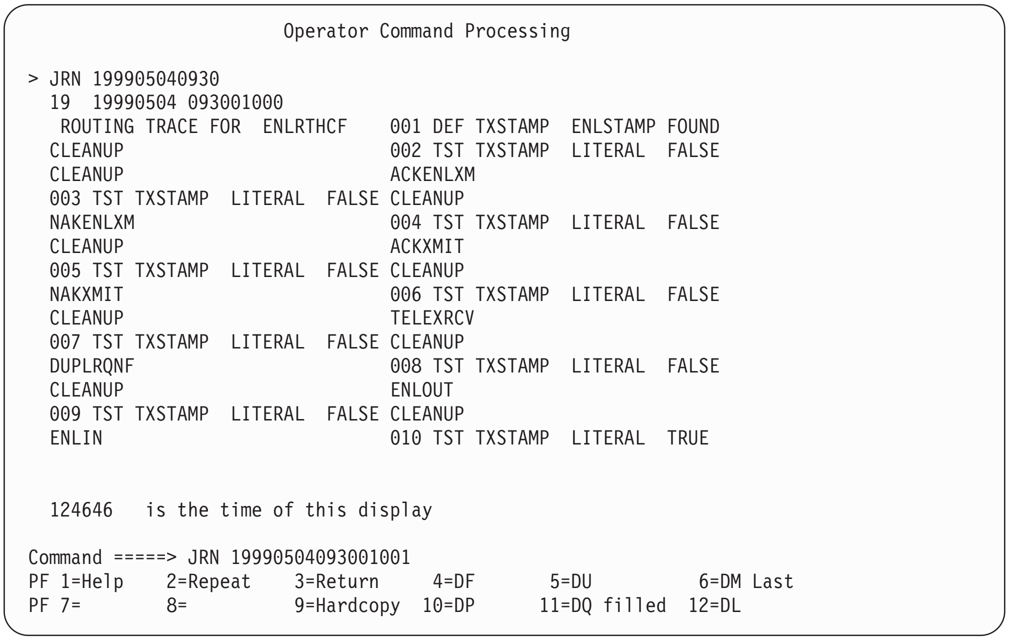
STATUS
: Enter the keyword **STATUS** to display the current status of the journal file.

Example

To display the first journal record written after 9:30 a.m. on 4 May 1999, enter:

```
jrn 199905040930
```

The journal record is displayed, with the command line containing the **jrn** command for accessing the next consecutive journal record, as shown in Figure 15 on page 74.



```
                        Operator Command Processing

> JRN 199905040930
  19  19990504 093001000
   ROUTING TRACE FOR  ENLRTHCF    001 DEF TXSTAMP  ENLSTAMP FOUND
  CLEANUP                         002 TST TXSTAMP  LITERAL  FALSE
  CLEANUP                         ACKENLXM
  003 TST TXSTAMP  LITERAL  FALSE CLEANUP
  NAKENLXM                        004 TST TXSTAMP  LITERAL  FALSE
  CLEANUP                         ACKXMIT
  005 TST TXSTAMP  LITERAL  FALSE CLEANUP
  NAKXMIT                         006 TST TXSTAMP  LITERAL  FALSE
  CLEANUP                         TELEXRCV
  007 TST TXSTAMP  LITERAL  FALSE CLEANUP
  DUPLRQNF                        008 TST TXSTAMP  LITERAL  FALSE
  CLEANUP                         ENLOUT
  009 TST TXSTAMP  LITERAL  FALSE CLEANUP
  ENLIN                           010 TST TXSTAMP  LITERAL  TRUE


  124646   is the time of this display

Command =====> JRN 19990504093001001
PF 1=Help    2=Repeat    3=Return     4=DF       5=DU          6=DM Last
PF 7=        8=          9=Hardcopy  10=DP      11=DQ filled  12=DL
```

Figure 15. Example of the Journal Display (jrn) Command Response

Chapter 13. DSLCIVAL—Customization and Installation Validation Program

DSLCIVAL is a program for cross-checking MERVA ESA installation and customization modules and parameters. The checks are performed in the following areas of MERVA ESA:

- Base Functions
- SWIFT Link
- Telex Link via a fault-tolerant system

For the following areas of MERVA ESA, installation is not verified by DSLCIVAL, but is verified by separate installation verification tests described in the chapters shown:

MERVA Link
"Chapter 14. Verifying the Installation of MERVA Link" on page 79

| **FMT/ESA using MERVA Link**
| "Chapter 15. Verifying the Installation of FMT/ESA Using MERVA Link" on
| page 85

MERVA-MQI Attachment
"Chapter 16. Verifying the Installation of MERVA-MQI Attachment" on page 89

| **FMT/ESA using MERVA-MQI Attachment**
| "Chapter 17. Verifying the Installation of FMT/ESA Using MERVA-MQI
| Attachment" on page 103

DSLCIVAL is executed as a batch program, or for MERVA ESA under CICS, as a CICS transaction. Both the batch program and the transaction are executed in the same environment as MERVA ESA so you must use the same load libraries. When errors are encountered, processing continues as long as possible and each determined error or problem results in an information message. For the batch program, these messages are printed in the JOB console listing. For the CICS transaction, they are shown on the screen terminal and you can route them to a hardcopy printer.

DSLCIVAL does the following:

- Verifies that the required modules and tables are in the load libraries.
- Checks the consistency of the parameters used in various MERVA ESA tables. Because some of these parameter specifications (for example, queue and function names) are interdependent, it is not possible to check this during assembly time.

DSLCIVAL checks the following items:

- Existence of all required execution modules
- Existence of all required tables
- Existence of all referenced routing modules
- Existence of loadable user exits
- Existence of required MCBs
- JRNBUF specification with respect to NICBUF
- Function table: Check cross reference queue names

- MFS Program table: Load MFS exits, PFK tables and MCBs
- Message Type table: Load MCBs
- SWIFT Link: Existence of DWSPRM
- SWIFT Link: Existence of DWSLTT
- SWIFT Link: LTT ready queues in function table
- SWIFT Link: Check existence of routing modules
- Telex Link: Existence of ENLPRM
- Telex Link via a fault-tolerant system: Existence of telex functions
- Telex Link via a fault-tolerant system: Telex Buffer Size and NICBUF
- Telex Link via a fault-tolerant system: Key (length, offset, field name) definitions of telex functions

The following checks apply to only the CICS transaction:

- Check for CICS program definitions for MERVA ESA load modules
- Verify MERVA ESA file table against CICS VSAM files
- Function table: Check existence of printer terminals in CICS
- Function table: Check existence of transactions in CICS

Customization and Installation Validation Batch Program

Here you find examples of the job-control statements you need to run the customization and installation validation batch program DSLCIVAL under MVS or VSE. You start this program on request.

Note: You must use the same libraries as the MERVA ESA applications use. Otherwise the resulting warning or error messages are meaningless.

The batch program requires at least 4MB virtual storage to run. MERVA ESA need not be active when the program is run.

Running the Batch Program under MVS

The JCL required to run the batch program DSLCIVAL under MVS is:

```
//.......  JOB .........
//RUN      EXEC PGM=DSLCIVAL,REGION=4096K
//STEPLIB  DD DSN=loadlib,DISP=SHR
//SYSUDUMP DD SYSOUT=*
//DSLSNAP  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

where **loadlib** represents the library containing the MERVA ESA load modules.

Running the Batch Programs under VSE

The JCL required to run the batch program DSLCIVAL under VSE is:

```
// JOB  .........
// ASSGN SYSLST,00E
// DLBL library,'program library',99/365,SD
// EXTENT ,volid
LIBDEF *,SEARCH=(library.sublib, ....)
// OPTION PARTDUMP
// EXEC DSLCIVAL,SIZE=300K
/*
/&
```

where:

volid The identification of the volume containing the program library

program library The name of the library containing the MERVA ESA programs

library.sublib The name of the program sublibrary which contains the MERVA ESA programs (you can code a list of sublibraries)

Customization and Installation Validation as a CICS Transaction

To start MERVA ESA under CICS/MVS or CICS/VSE, follow these steps:

1. Start CICS. You must use the same libraries as the MERVA ESA applications use in your CICS as the resulting warning or error messages are otherwise meaningless.
2. Logon to CICS on an IBM 327x Display Station that is defined as a CICS terminal.
3. Start the customization and installation validation transaction by entering the transaction code DSLC. If this transaction code has been changed in your installation, ask your system administrator for the transaction code defined for your installation. The transaction code is defined in the CICS transaction definitions for the program DSLCIVAC.

On successful startup of the MERVA ESA validation transaction the following message is displayed at the screen terminal:

```
DSL4000 MERVA ESA INSTALLATION CHECKING STARTED
```

Depending on the result of the checking process, error or warning messages are displayed on the screen terminal. When a screen terminal page has been filled, the following line is displayed:

```
... Press Enter to continue
```

The program waits until the ENTER key is pressed. Then the screen terminal page is cleared and more messages are displayed. The last message to be displayed is:

```
DSL4095 MERVA ESA Installation checking complete
```

Printing Messages of DSLCIVAC

If you need a hardcopy of the warning and error messages of DSLCIVAC, you can specify a printer terminal at the beginning when entering the transaction code. All warning and error messages, for example, are routed to the printer with the logical terminal name *pppp* when you enter the following on a CICS screen terminal:

```
DSLC PRINT=pppp
```

Chapter 14. Verifying the Installation of MERVA Link

To verify the installation of MERVA Link you must run a separate MERVA Link installation verification test. This test takes place when the MERVA Link is installed and MERVA ESA and your operating system have been upgraded.

Note: Make sure that the installation verification test is passed successfully before you start customizing the system according to your needs.

Special Test Conditions

To run the MERVA Link installation verification test, the MERVA ESA test commands (see "Chapter 12. The MERVA ESA Queue Test and Journal Display Commands" on page 69) must be enabled in your system. The MERVA Link installation verification test runs in a local MERVA ESA system in a back-to-back environment: The MERVA Link sends a message to itself, and notifies itself that it has received that message. A real remote MERVA ESA partner system is not involved in this installation verification test.

All MERVA Link programs required in an intersystem connection are verified. The test message, however, is not transmitted via an intersystem communication line, and the connection of the MERVA Link to the intersystem communication facilities of CICS or IMS are not verified.

The connection of the MERVA Link to the intersystem communication facilities of your operating system is described in the *MERVA for ESA Customization Guide*.

To run the installation verification test of the MERVA Link, you must assume the role of the following applications in your local MERVA ESA system:

- An application, which edits a message and sends this message to the partner MERVA ESA system.
- The partner application, which processes that message and returns an acknowledgment message to the sender of the original message.
- The sending application, which must verify that it has received the acknowledgment and that the message has been routed as a completed message.

Note: Each step described in the following installation verification scenario is usually done by an application and not by a user. To run this test you must be authorized to work with message type DEMO (the MERVA Link sample message).

You must allow the user ID to be used in the test scenario to access the following functions:

EKAEOM	Edit Outgoing Message Queue
EKAEAM	Edit Acknowledgment Message Queue
EKACMQ	Completed Message Queue
EKARMQ	Received Message Queue

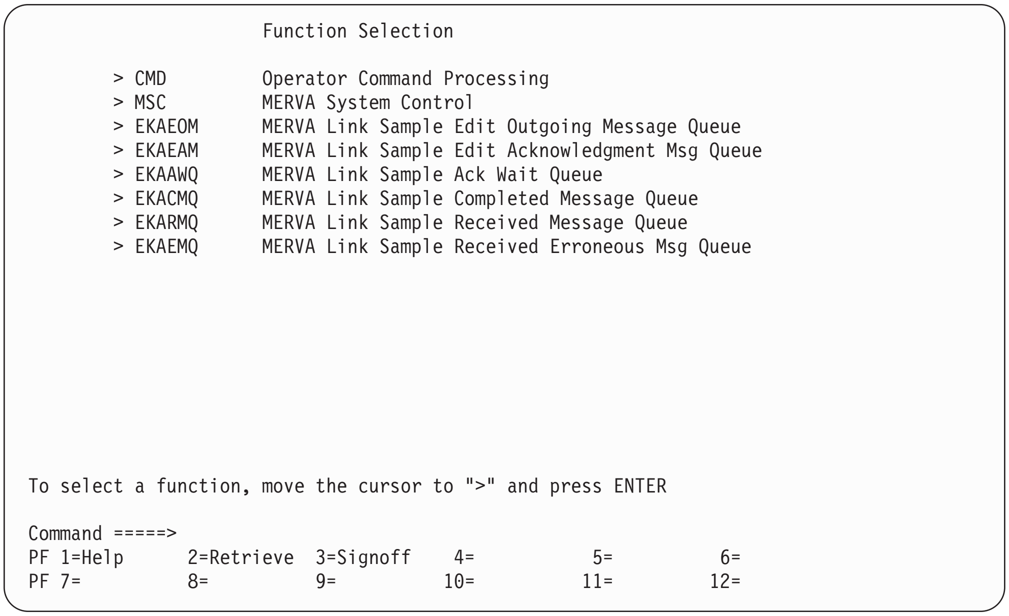
During the sample scenario you need not have access to the functions MSC, EKAAWQ, and EKAEMQ. However, to become familiar with the MERVA Link, it is useful to have access to MSC.

Carry out the following steps to verify your MERVA Link installation using the sample scenario provided.

Step 1: Edit and Send a Message

To edit and send a message using MERVA Link you must:

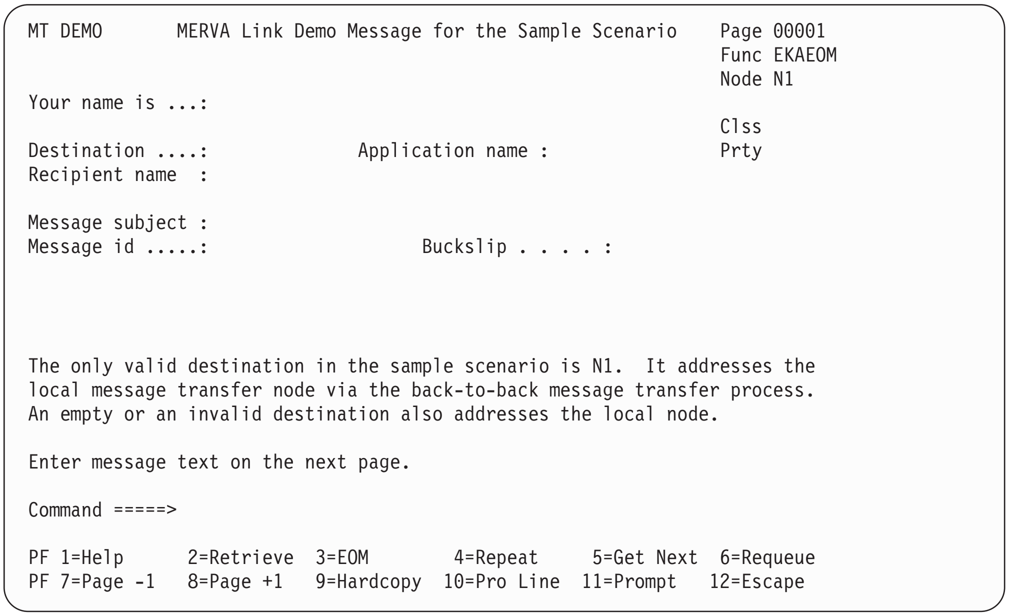
1. Sign on to MERVA ESA as the MERVA ESA user who runs the installation and verification test for MERVA Link.
2. Select EKAEOM, the sample edit outgoing message queue, in the Function Selection menu, shown in Figure 16.



```
                        Function Selection

          > CMD          Operator Command Processing
          > MSC          MERVA System Control
          > EKAEOM       MERVA Link Sample Edit Outgoing Message Queue
          > EKAEAM       MERVA Link Sample Edit Acknowledgment Msg Queue
          > EKAAWQ       MERVA Link Sample Ack Wait Queue
          > EKACMQ       MERVA Link Sample Completed Message Queue
          > EKARMQ       MERVA Link Sample Received Message Queue
          > EKAEMQ       MERVA Link Sample Received Erroneous Msg Queue









To select a function, move the cursor to ">" and press ENTER

Command =====>
PF 1=Help      2=Retrieve  3=Signoff    4=          5=         6=
PF 7=          8=          9=          10=         11=        12=
```

Figure 16. Selecting a MERVA Link Sample Queue

3. To edit a sample message of the type DEMO (shown in Figure 17 on page 81), enter **mt demo** on the command line of the panel displayed for function EKAEOM.



```
MT DEMO       MERVA Link Demo Message for the Sample Scenario    Page 00001
                                                                 Func EKAEOM
                                                                 Node N1
Your name is ...:
                                                                 Clss
Destination ....:              Application name :                Prty
Recipient name  :

Message subject :
Message id .....:                    Buckslip . . . . :




The only valid destination in the sample scenario is N1.  It addresses the
local message transfer node via the back-to-back message transfer process.
An empty or an invalid destination also addresses the local node.

Enter message text on the next page.

Command =====>

PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 17. Sample Message of Type DEMO

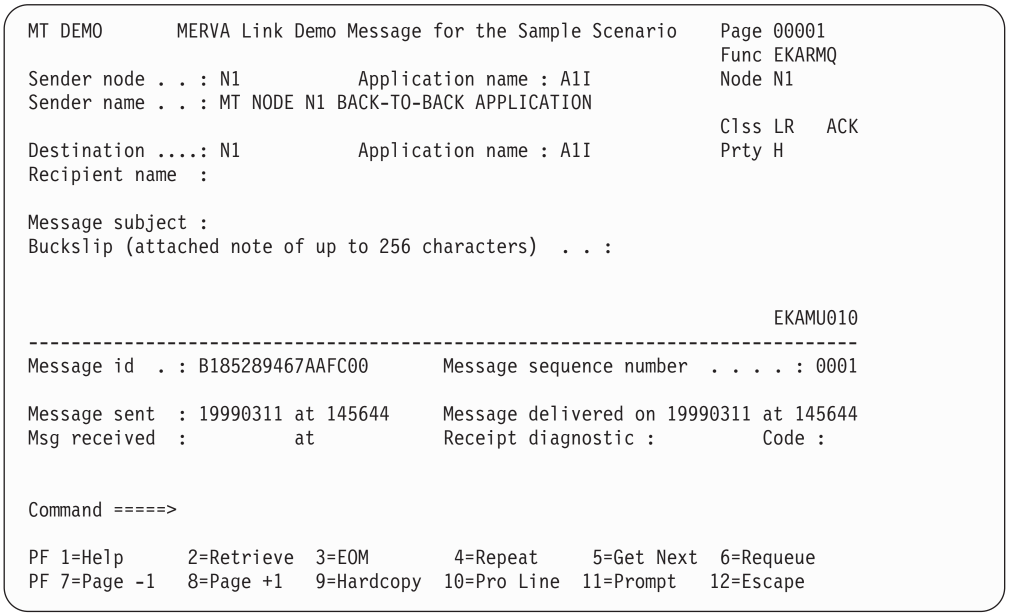
4. Enter some sample message information, for example, your name and address in the sample screen. Note, however, that some fields are protected. Press PF8 to scroll to the second page of the message. On the second page of the demo message you can enter text in free format.

   If you do not specify a priority, MERVA Link sets it to H for high priority if the applicable send queue was specified in the first place in the send queue cluster.
5. Press PF3 to finish data entry and send the message to the partner application.
6. Press PF3 again to return to the Function Selection menu.

Step 2: Process the Message and Send an Acknowledgment

To assume the role of the receiving application you need not sign off from MERVA ESA. To process a message and send an acknowledgment, you must:

1. Select EKARMQ, the sample received message queue, in the Function Selection menu.
2. Press PF5 to display the message you have received from your partner application. This message contains all data entered in “Step 1: Edit and Send a Message” on page 80 and all defaults provided by the MERVA Link. Figure 18 on page 82 shows the sample message that is displayed if you did not enter any data previously.



```
MT DEMO       MERVA Link Demo Message for the Sample Scenario    Page 00001
                                                                 Func EKARMQ
Sender node . . : N1           Application name : A1I            Node N1
Sender name . . : MT NODE N1 BACK-TO-BACK APPLICATION
                                                                 Clss LR   ACK
Destination ....: N1           Application name : A1I            Prty H
Recipient name  :

Message subject :
Buckslip (attached note of up to 256 characters)  . . :


                                                                      EKAMU010
------------------------------------------------------------------------------
Message id  . : B185289467AAFC00       Message sequence number  . . . . : 0001

Message sent  : 19990311 at 145644     Message delivered on 19990311 at 145644
Msg received  :          at            Receipt diagnostic :          Code :


Command =====>

PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 18. Received Sample Message of Type DEMO

In the panel shown in Figure 18:

- The value in the message class field (**Clss**) is **LR** for last received message.
- The letters **ACK** following the message class indicate that the sending application asks for an acknowledgment of this message.
- The value of the message priority field (**Prty**) is **H** for high priority.
- The MERVA ESA MFS user exit **EKAMU010** associated with the message transfer application A1I has left its signature at the end of the buckslip. This indicates that it was active for a received message.

Notes:

a. The message identifier and the date and time will differ for your panel. The message sequence number can also be different. If you run this sample for the first time, it is 0001 as shown in Figure 18.

b. If in the DSLPRM the value of UMR=YES, the MERVA ESA Unique Message Reference (UMR) is used as MERVA Link message identifier, and the UMR is shown in an additional line in the TOP frame of your display.

3. When you have read the received message, processing of the received message is completed. Enter the **del** command to delete this message.
4. Press PF3 in the Message Selection menu to return to the Function Selection menu.
5. Select EKAEAM, the sample edit acknowledgment queue, in the Function Selection menu to edit an acknowledgment for the message you received.
6. Press PF5 to display the received message in the EKAEAM queue. The received message now serves as a skeleton for editing an acknowledgment message. The received message already contains information required by an acknowledgment message, such as originator and recipient information and the identifier of the message which is to be acknowledged.
7. To route the completed message to another queue, the sample scenario requires valid receipt information, that is, a valid receipt return code or specific information at the beginning of the Message subject field.

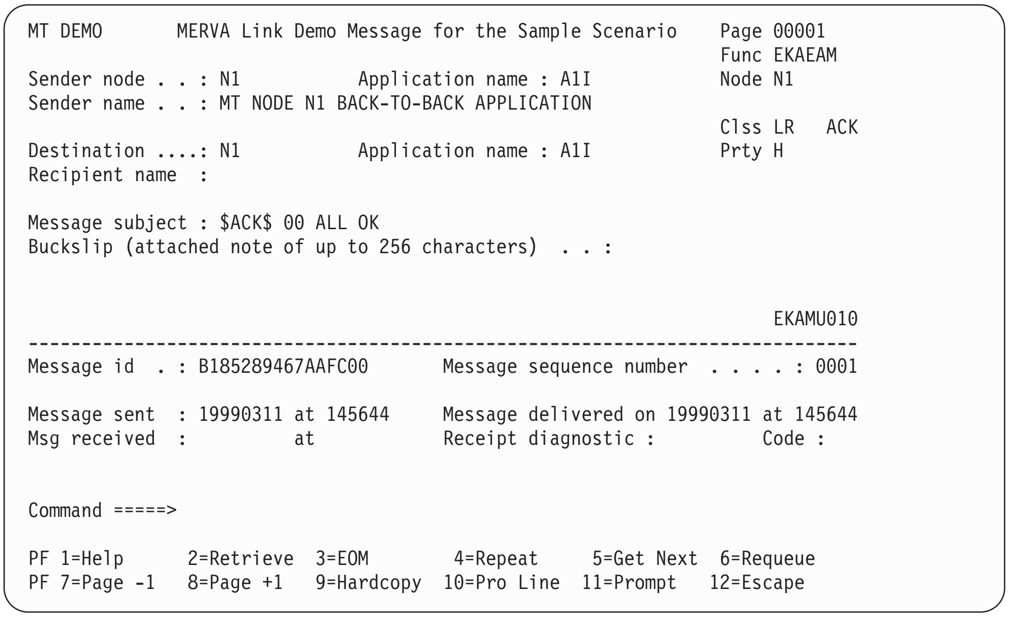
Valid receipt information in the Message subject field is:

- ACK 00
- ACK 04
- ACK 08

Note: The ACK indicator ACK is separated from the numeric return code by a blank. The message subject is followed by a receipt diagnostic code of 6 characters.

Enter the following data at the beginning of the field Message subject, as shown in Figure 19:

```
$ACK$ 00 ALL OK
```



```
MT DEMO       MERVA Link Demo Message for the Sample Scenario    Page 00001
                                                                 Func EKAEAM
Sender node . . : N1          Application name : A1I             Node N1
Sender name . . : MT NODE N1 BACK-TO-BACK APPLICATION
                                                                 Clss LR   ACK
Destination ....: N1          Application name : A1I             Prty H
Recipient name  :

Message subject : $ACK$ 00 ALL OK
Buckslip (attached note of up to 256 characters)  . . :


                                                                      EKAMU010
------------------------------------------------------------------------------
Message id  . : B185289467AAFC00       Message sequence number  . . . . : 0001

Message sent  : 19990311 at 145644     Message delivered on 19990311 at 145644
Msg received  :          at            Receipt diagnostic :         Code :


Command =====>

PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 19. Send Acknowledgment Message

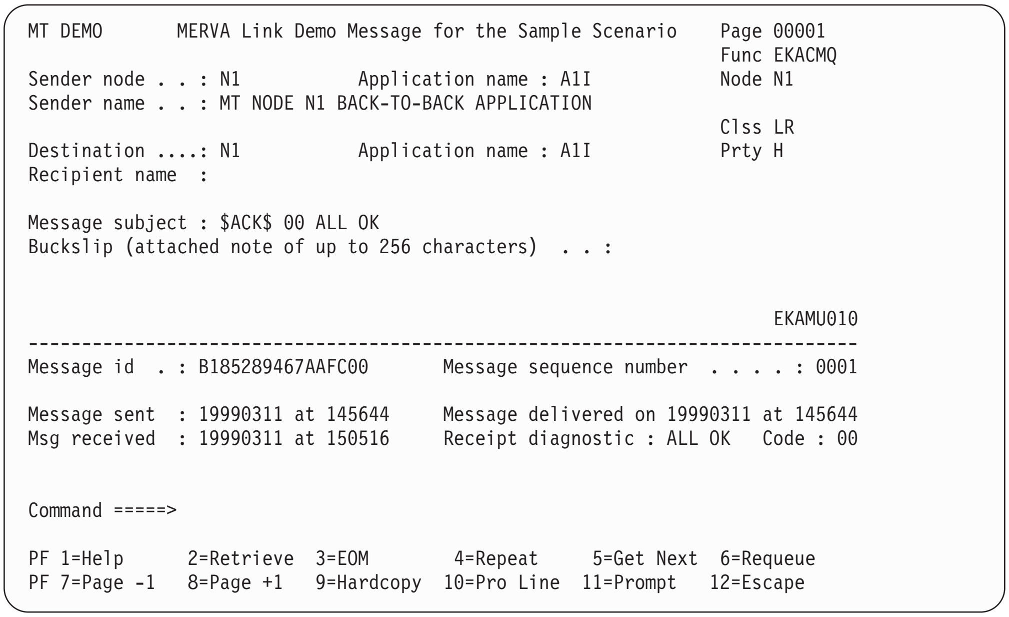
8. Press PF3 (EOM) to send the acknowledgment message to your partner application which sent the original message.
9. Press PF3 to return to the Function Selection menu.

Step 3: Verify the Receipt of the Acknowledgment

Now you assume the role of the sending application again. To verify that you have received the acknowledgment and that the message has been routed as a completed message you must:

1. Select EKACMQ, the completed message queue, in the Function Selection menu.
2. Press PF5 to display the completed message, which is the original message with acknowledgment information. The acknowledgment information contains the date and time, the receipt diagnostic code (if specified), and the mandatory receipt return code (00 in this example). If you have entered a new message subject or a new buckslip in the acknowledgment message, these data elements are also shown, replacing the original data elements.

Figure 20 shows a sample of a completed message, where you entered ACK 00 ALL OK in the Message subject field as receipt information.



```
MT DEMO       MERVA Link Demo Message for the Sample Scenario    Page 00001
                                                                 Func EKACMQ
Sender node . . : N1          Application name : A1I             Node N1
Sender name . . : MT NODE N1 BACK-TO-BACK APPLICATION
                                                                 Clss LR
Destination ....: N1          Application name : A1I             Prty H
Recipient name  :

Message subject : $ACK$ 00 ALL OK
Buckslip (attached note of up to 256 characters)  . . :


                                                                      EKAMU010
------------------------------------------------------------------------------
Message id  . : B185289467AAFC00       Message sequence number  . . . . : 0001

Message sent  : 19990311 at 145644     Message delivered on 19990311 at 145644
Msg received  : 19990311 at 150516     Receipt diagnostic : ALL OK   Code : 00


Command =====>

PF 1=Help      2=Retrieve  3=EOM        4=Repeat     5=Get Next  6=Requeue
PF 7=Page -1   8=Page +1   9=Hardcopy  10=Pro Line  11=Prompt   12=Escape
```

Figure 20. Completed Sample Message Alternative (with Receipt Information Entered)

3. Message processing in this sample scenario is completed when you have read the message in the completed message queue. To delete this message enter the **del** command.
4. Press PF3 in the Message Selection menu to return to the Function Selection menu.

When you have finished the installation verification test, you have:

- Installed MERVA Link in your operating system (CICS or IMS)
- Verified the basic functions of the MERVA Link

The communication between the MERVA Links installed in different remote MERVA ESA systems also requires the installation and customization of the intercommunication facilities of your operating system and the backbone network with the required hardware. Refer to the appropriate CICS and IMS documentation and related books for details. The *MERVA for ESA Customization Guide* contains examples of how to customize the MERVA Link and the intercommunication facilities of the operating system to run according to your requirements.

Chapter 15. Verifying the Installation of FMT/ESA Using MERVA Link

The MERVA Link Back-to-Back Test Environment provides a means to test FMT/ESA.

The message-sending and the message-receiving side can be executed in the same MERVA ESA system both under CICS and under IMS. A connection to a remote system is not required in this environment.

In the Back-to-Back Test Environment, messages are sent and received using **the same ASP**. The ASP entry specifies its own address (NODE, ASP NAME) as the address of the destination application.

The MERVA Link sample partner table to establish the Back-to-Back Test Environment is shown below:

```
         TITLE 'MERVA LINK SAMPLE PARTNER TABLE FOR NODE N1'
*----------------------------------------------------------------------
* THIS PARTNER TABLE CAN BE USED IN THE INSTALLATION VERIFICATION FOR
* THE MERVA-TO-MERVA FINANCIAL MESSAGE TRANSFER/ESA (FMT/ESA) FUNCTION.
* IT IS VALID BOTH IN A CICS AND IN AN IMS ENVIRONMENT.
*----------------------------------------------------------------------
*
         EKAPT TYPE=INITIAL,NODE=N1    MESSAGE TRANSFER NODE N1
*
*----------------------------------------------------------------------
* A1I    N1    BACK-TO-BACK ASP
*----------------------------------------------------------------------
         EKAPT TYPE=ASP,               APPLICATION SUPPORT PROCESS     *
               NAME=(A1I,'FMT/ESA INSTALLATION VERIFICATION'),         *
               DEST=(N1,A1I),          DESTINATION ADDRESS             *
               MFSEXIT=7044,           FMT/ESA USER EXIT NUMBER        *
               SENDQC=EKASWSND,        FMT/ESA SEND QUEUE              *
               CONTROL=EKASIMCQ,       FMT/ESA CONTROL QUEUE           *
               IRROUTE=(ACK,EKASWAWQ,CTLQ), RECEIPT REPORT CORRELATION *
               CONFIRM=NO,             ACK GENERATION IN SENDING MERVA *
               FORMAT=(,,W),           SWIFT LINE FORMAT IDENTIFIER    *
               MTP=BTB                 NAME OF APPLICABLE MTP
*
*----------------------------------------------------------------------
* BTB          BACK-TO-BACK MTP
*----------------------------------------------------------------------
         EKAPT TYPE=MTP,               MESSAGE TRANSFER PROCESS        *
               NAME=BTB,               INTERNAL MTP NAME               *
               DEST=BTB,               SYNCHRONOUS BACK-TO-BACK CONN   *
               ASP=A1I                 NAME OF APPLICABLE ASP
*
         EKAPT TYPE=FINAL              GENERATE PARTNER TABLE TRAILER
         END
```

Parameter DEST=(N1,A1I) shows the same destination address as specified in parameters NODE=N1 and NAME=(A1I,...).

In order to verify the FMT/ESA installation, do the following:

1. Install partner table EKAPTFMT.

   In the source library you find the sample partner table EKAPTFMT. Assemble and link-edit the partner table as EKAPT. The machine-readable material for the MERVA Link contains the JCL to install a partner table as a USERMOD, for example the jobs EKAPTEYA or EKAPTEYB.

In MVS, the MERVA ESA source library is a partitioned data set with the low-level qualifier SDSLSRC0. In VSE, the sample partner table EKAPTFMT is part of the source library.

2. Start up your MERVA ESA system and sign on to MERVA ESA.
3. Create a new MERVA ESA user MASSIM via user file maintenance.

   Assign the following functions to MASSIM. The FMT/ESA functions start with EKA. They are described in the *MERVA for ESA Customization Guide*:
   - EKASWDE0
   - EKASWAI0
   - EKASWSND
   - EKASWVE0
   - EKASWACK
   - EKASWSDO
   - EKASWAO0
   - EKASWLEQ
   - EKASWREQ
   - EKASIMCQ
   - EKAISNCQ
   - EKAOSNCQ
   - CMD
   - MSC
   - AUT

   Assign at least message type S100 to MASSIM.
4. Sign off from MERVA ESA.
5. Sign on to MERVA ESA with the new user ID MASSIM.
6. Create both a sending and a receiving authenticator key via authenticator-key file maintenance function AUT.
7. Select data entry queue EKASWDE0 from the Function Selection panel.
8. Enter in the command line: MT S100, press the ENTER key.
9. Enter a SWIFT input message in queue EKASWDE0, press PF key 3 (EOM).
10. Enter in the command line: return EKASWAI0, press the ENTER key.
11. In EKASWAI0, press PF key 6 (Get First).
12. You see the SWIFT input message you have just entered. Press PF key 3 (EOM). The message moves to the MERVA Link send queue EKASWSND and message transmission is initiated.

You can monitor the message transmission on one of the MERVA Link control panels. You get to these panels when you select the function MSC from the Function Selection panel.

When message transmission has ended, inspect the appropriate queues of the FMT/ESA sample scenario. The following results are expected:
- EKASWACK contains the acknowledged SWIFT input message.
- EKASWSDO contains the generated SWIFT output message.
- EKAISNCQ contains the ISN '000001' for ASP 'A1I'.

- EKAOSNCQ contains the OSN '000001'.

Note: In case the SWIFT authentication initialization program DWSAUTIN was not started, EKASWVE0 contains the SWIFT input message with an error message. Then proceed as follows:

1. Enter in the command line of the CMD function: **start xx**. xx can be either the descriptive name of DWSAUTIN as defined in the MERVA ESA Nucleus Program Table DSLNPTT, or the program identification of DWSAUTIN which is generated in DSLNPTT. Press the ENTER key.
2. Enter in the command line: return EKASWVE0, press the ENTER key.
3. In EKASWVE0, press PF key 6 (Get First).
4. You see the SWIFT input message with the error message. Press PF key 3 (EOM). The message moves to the authorization queue EKASWAI0.
5. Enter in the command line: return EKASWAI0, press the ENTER key.
6. In EKASWAI0, press PF key 6 (Get First).
7. You see the SWIFT input message with the error message. Press PF key 3 (EOM). The message moves to the MERVA Link send queue EKASWSND and message transmission is initiated.

After message transmission, you should find the messages as described above in the queues EKASWACK, EKASWSDO, EKAISNCQ, and EKAOSNCQ.

Chapter 16. Verifying the Installation of MERVA-MQI Attachment

Before you can verify whether the MERVA-MQI Attachment was installed correctly, the following prerequisites must be satisfied:

- For MVS:
  - The MQSeries must have been started.
  - Depending on the DC system used, either the MQSeries-supplied CICS task initiator CKTI or the IMS trigger monitor must have been started.
  - In a CICS installation, the JCL in the startup job must contain:
    - The MQSeries libraries in the STEPLIB and DFHRPL concatenation.
    - The INITPARM parameter in the CICS system initialization table (SIT), or the SYSIN override.

    See the program directory for MERVA ESA for details.
  - In an IMS installation, the JCL in the MPP job must contain the MQSeries libraries in the STEPLIB or DFSESL concatenation. Optionally, the JCL in the MPP can contain the SSM EXEC parameter to identify a subsystem member (SSM) in the IMS.PROCLIB library.

    See the program directory for MERVA ESA for details.
- For VSE:
  - The JCL in the CICS startup job must contain the MQSeries library in the LIBDEF search chain, and the label definitions for the MQSeries data sets.

    See the program directory for MERVA ESA for details.
  - The MQSeries environment must have been set up and initialized.

    There are several ways of doing this:
    - On the CICS terminal, enter **MQSE**

      This sets up the MQSeries environment. After the setup is complete, initialize the MQSeries queue manager by doing one of:
      - On the CICS terminal, type **MQIT**
      - On the CICS terminal, type **MQMT** to display the Master Terminal Main Menu panel, then:
        - Select 2 - Operations
        - Select 4 - Initialization/Shutdown
        - Type **I** in the function field and press PF6
    - On the CICS terminal, enter **MQSE I**

      This combines the environment setup and initialization.

    See the *MQSeries for VSE/ESA System Management Guide* for details.

Local MQI queues are used for the installation verification. Therefore the distributed queue management (DQM) of MQSeries is not required.

Installation Verification Scenario

The scenario to verify the installation comprises two MERVA-to-MQI send processes and two MQI-to-MERVA receive processes. The following send processes are used:

SPROC1 Sends MQI request messages.

SPROC2 Sends MQI reply messages.

The following receive processes are used:

RPROC1 Receives MQI reply and report messages for the request messages.

RPROC2 Receives MQI request messages and, for MVS, MQI report messages for the reply messages.

The send and receive processes are defined in the MERVA-MQI Attachment sample process table DSLKPSAM (MVS) or DSLKPSMV (VSE), and are shown in the following figures.

```
DSLKPROC TYPE=SEND,                                    *
      NAME=SPROC1,                                     *
      ALLSNDQ=((DSLMRSQ1,DSL.MQI.SNDRCV)),             *
      COAWQ=DSLMRCOA,                                  *
      CODWQ=DSLMRCOD,                                  *
      ACKWQ=DSLMRAWQ,                                  *
      MQICTLQ=DSL.MQI.CONTROL,                         *
      MRVCTLQ=(DSLMRCQS,CONTINUE),                     *
      MRVSTAQ=DSLMRSTS,                                *
      REPLYTQ=DSL.MQI.REPLY_TO_Q,                      *
      EXIT=8001,                                       *
      MQFMT=BLANK,                                     *
      OPMSDM=SUBSET
```

Figure 21. Send Process SPROC1

```
DSLKPROC TYPE=SEND,                                    *
      NAME=SPROC2,                                     *
      ALLSNDQ=((DSLMRSQ2,DSL.MQI.SNDRCV)),             *
      COAWQ=DSLMRCOA,                                  *
      CODWQ=DSLMRCOD,                                  *
      MQICTLQ=DSL.MQI.CONTROL,                         *
      MRVCTLQ=(DSLMRCQS,CONTINUE),                     *
      MRVSTAQ=DSLMRSTS,                                *
      REPLYTQ=DSL.MQI.SNDRCV,                          *
      EXIT=8001,                                       *
      MQFMT=BLANK,                                     *
      OPMSDM=SUBSET
```

Figure 22. Send Process SPROC2 for MVS

```
DSLKPROC TYPE=SEND,                                    *
      NAME=SPROC2,                                     *
      ALLSNDQ=((DSLMRSQ2,DSL.MQI.SNDRCV)),             *
      MQICTLQ=DSL.MQI.CQ,                              *
      MRVCTLQ=(DSLMRCQS,CONTINUE),                     *
      MRVSTAQ=DSLMRSTS,                                *
      EXIT=8001,                                       *
      MQFMT=BLANK,                                     *
      OPMSDM=SUBSET
```

Figure 23. Send Process SPROC2 for VSE

```
DSLKPROC TYPE=RECEIVE,                                *
      NAME=RPROC1,                                    *
      MQIRCVQ=(DSL.MQI.REPLY_TO_Q),                   *
      MRVCTLQ=DSLMRCQR,                               *
      MRVSTAQ=DSLMRSTR,                               *
      EXIT=8002,                                      *
      OPMSDM=SUBSET
```

Figure 24. Receive Process RPROC1

```
DSLKPROC TYPE=RECEIVE,                                *
      NAME=RPROC2,                                    *
      MQIRCVQ=(DSL.MQI.SNDRCV),                       *
      MRVCTLQ=DSLMRCQ1,                               *
      MRVSTAQ=DSLMRSTR,                               *
      EXIT=8002,                                      *
      OPMSDM=SUBSET
```

Figure 25. Receive Process RPROC2

MQI Queues

As the scenario does not require the DQM of MQSeries, channel and remote queue definitions are not needed. The following local MQI queues are used for the scenario:

DSL.MQI.CONTROL (control queue)

The queue is used as a control queue in the send process SPROC1. For MVS, it is also used in the send process SPROC2.

DSL.MQI.CQ (second control queue; VSE only)

This queue is used as a control queue in the send process SPROC2 (VSE only).

DSL.MQI.REPLY_TO_Q (reply-to queue)

The queue is specified as a reply-to queue in the send process SPROC1. It is used as a receive queue for the incoming MQI reply and report messages in the receive process RPROC1.

DSL.MQI.SNDRCV (send and receive queue)

The queue is used as a send queue for the outgoing MQI request messages in the send process SPROC1. It is specified, but not used, as a send queue in the send process SPROC2.

For MVS, it is specified as a reply-to queue in the send process SPROC2. It is used as a receive queue for the incoming MQI request messages and, for MVS, report messages in the receive process RPROC2.

Defining Queues in MQSeries for MVS/ESA

The complete set of MQI queues, processes, and channels is provided in the copy books DSLKCSQC (for CICS) and DSLKCSQI (for IMS). Refer to these copy books for details on the parameters specified in the MQSeries DEFINE commands.

Some of the parameter values in these copy books must be modified before the resources can be defined to MQSeries in your installation. The changes that must be made are indicated in the copy books. For example, you will definitely need to set a new value for the initiation queue (parameter INITQ), which is used in the definition of the queue DSL.MQI.RECEIVE. The queues DSL.MQI.REPLY_TO_Q and DSL.MQI.SNDRCV also refer to this initiation queue.

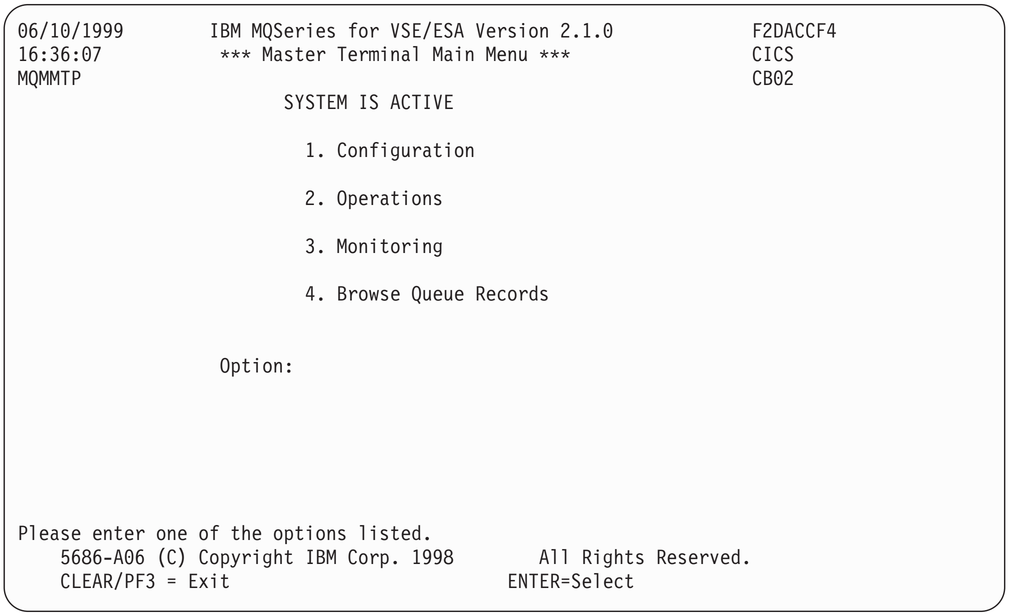
You can use the appropriate copy book as input for the MQSeries utility program CSQUTIL. The program directory for MERVA ESA contains a sample job for CSQUTIL.

If you are using MQSeries for MVS/ESA, you can continue with "MERVA ESA Queues" on page 95.

Defining Queues in MQSeries for VSE/ESA

MQI resources, like queues and channels, are defined using the MQSeries master terminal transaction MQMT. For the installation verification scenario, four local queues have to be defined.

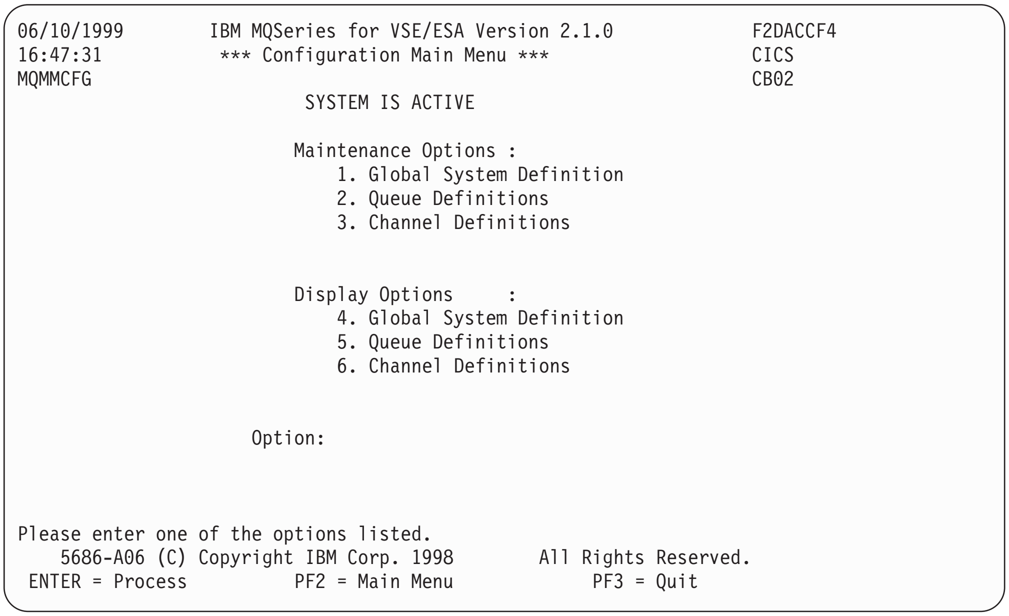
1. On the CICS terminal, enter the transaction code MQMT:



```
06/10/1999        IBM MQSeries for VSE/ESA Version 2.1.0             F2DACCF4
16:36:07           *** Master Terminal Main Menu ***                CICS
MQMMTP                                                               CB02
                         SYSTEM IS ACTIVE

                           1. Configuration

                           2. Operations

                           3. Monitoring

                           4. Browse Queue Records


                   Option:






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

Figure 26. Master Terminal Main Menu

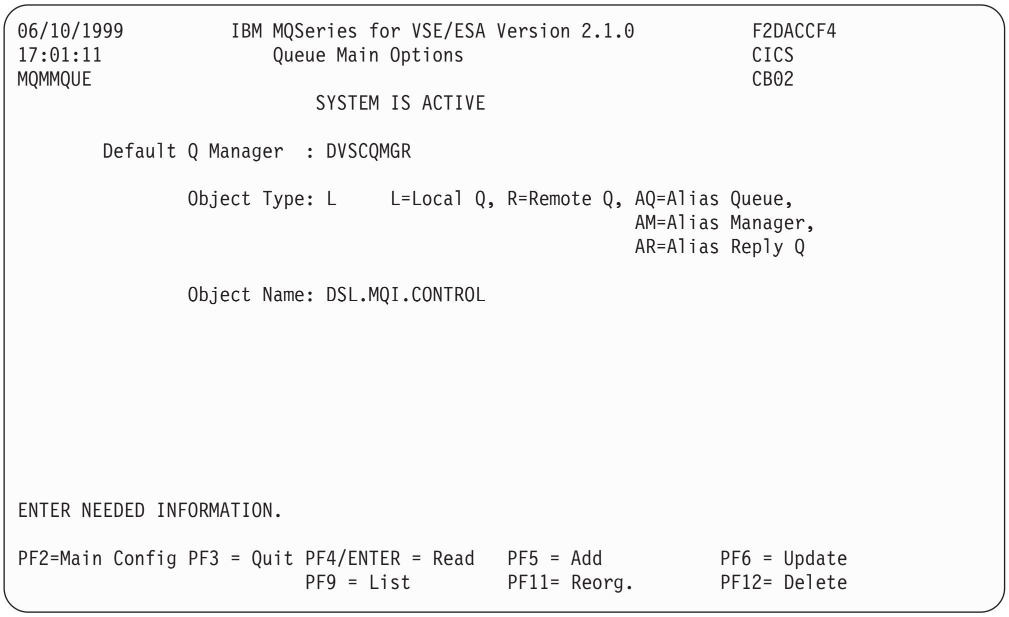
2. Select option 1 to configure.



```
06/10/1999       IBM MQSeries for VSE/ESA Version 2.1.0            F2DACCF4
16:47:31          *** Configuration Main Menu ***                  CICS
MQMMCFG                                                            CB02
                           SYSTEM IS ACTIVE

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


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


                      Option:



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

Figure 27. Configuration Main Menu

3. Select option 2 to work with queue definitions.

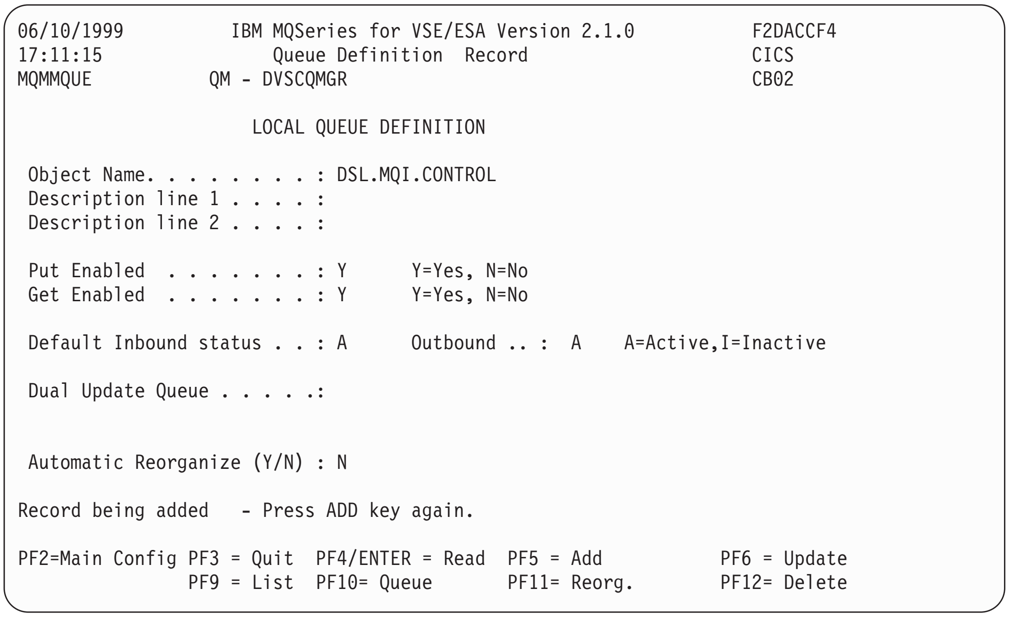


```
06/10/1999         IBM MQSeries for VSE/ESA Version 2.1.0          F2DACCF4
17:01:11               Queue Main Options                          CICS
MQMMQUE                                                            CB02
                            SYSTEM IS ACTIVE

        Default Q Manager  : DVSCQMGR

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

                Object Name: DSL.MQI.CONTROL








ENTER NEEDED INFORMATION.

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

Figure 28. Queue Main Options

4. Select an Object Type of L and specify the name of the queue as Object Name. Queue names are:
   - DSL.MQI.CONTROL
   - DSL.MQI.CQ
   - DSL.MQI.REPLY_TO_Q
   - DSL.MQI.SNDRCV

5. Press PF5.

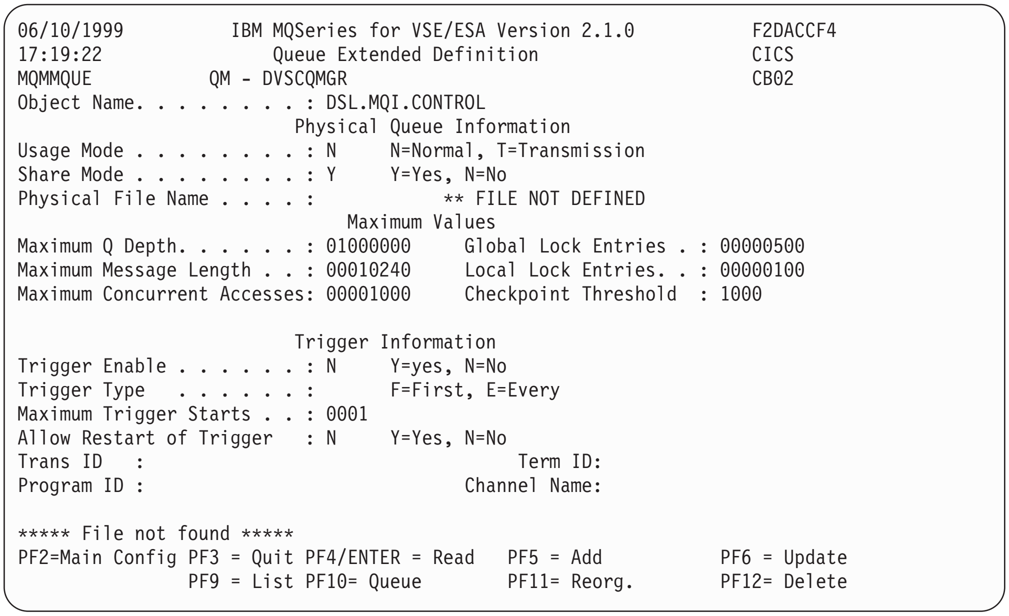


```
06/10/1999         IBM MQSeries for VSE/ESA Version 2.1.0          F2DACCF4
17:11:15               Queue Definition  Record                    CICS
MQMMQUE           QM - DVSCQMGR                                    CB02

                     LOCAL QUEUE DEFINITION

 Object Name. . . . . . . . : DSL.MQI.CONTROL
 Description line 1 . . . . :
 Description line 2 . . . . :

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

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

 Dual Update Queue . . . . .:


 Automatic Reorganize (Y/N) : N

Record being added   - Press ADD key again.

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

Figure 29. Queue Definition Record

For each queue name, accept the default values displayed on the panel.

6. Press PF5 again.



```
06/10/1999         IBM MQSeries for VSE/ESA Version 2.1.0          F2DACCF4
17:19:22               Queue Extended Definition                   CICS
MQMMQUE           QM - DVSCQMGR                                    CB02
Object Name. . . . . . . . : DSL.MQI.CONTROL
                          Physical Queue Information
Usage Mode . . . . . . . . : N     N=Normal, T=Transmission
Share Mode . . . . . . . . : Y     Y=Yes, N=No
Physical File Name . . . . :           ** FILE NOT DEFINED
                               Maximum Values
Maximum Q Depth. . . . . . : 01000000     Global Lock Entries . : 00000500
Maximum Message Length . . : 00010240     Local Lock Entries. . : 00000100
Maximum Concurrent Accesses: 00001000     Checkpoint Threshold  : 1000

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

***** File not found *****
PF2=Main Config PF3 = Quit PF4/ENTER = Read   PF5 = Add          PF6 = Update
                PF9 = List PF10= Queue        PF11= Reorg.       PF12= Delete
```

Figure 30. Queue Extended Definition

For each queue:

- Specify the name of a CICS file as Physical File Name to store messages for this queue.

Sample file names are MQFI001, MQFI002, or MQFI003.

- Specify a Usage Mode of N.

For the receive queues DSL.MQI.REPLY_TO_Q and DSL.MQI.SNDRCV:

- Set Trigger Enable to Y.
- Set Trigger Type to F.
- Set Allow Restart of Trigger to Y.
- Specify DSLR as the Trans ID.

7. Press PF5 again.
8. Press PF2.

Repeat steps 4 to 8 until all queues have been defined.

MERVA ESA Queues

The following MERVA ESA queues are defined in the MERVA ESA function table DSLFNTT. Their usage is determined by the definitions in the MERVA-MQI Attachment sample process table DSLKPSAM (MVS) or DSLKPSMV (VSE), and the logic in the sample routing table DSLKQRT.

DSLMRAWQ Acknowledgment Wait Queue
The queue is used in the send process SPROC1. It contains the MQI request messages waiting for an MQI reply message. The queue must be defined with KEY1 in the DSLFNTT. KEY1 represents the *MsgId* field of the MQI control block MQMD. It is contained in the TOF field DSLKMSID.

DSLMRCOA Confirm On Arrival (COA) Wait Queue
The queue is used in the send process SPROC1 and, for MVS, SPROC2. It contains the MQI request and, for MVS, reply messages waiting for an MQI COA report message. The queue must be defined with KEY1 in the DSLFNTT. KEY1 represents the *MsgId* field of the MQI control block MQMD. It is contained in the TOF field DSLKMSID.

DSLMRCOD Confirm On Delivery (COD) Wait Queue
The queue is used in the send process SPROC1 and, for MVS, SPROC2. It contains the MQI request and, for MVS, reply messages waiting for an MQI COD report message. The queue must be defined with KEY1 in the DSLFNTT. KEY1 represents the *MsgId* field of the MQI control block MQMD. It is contained in the TOF field DSLKMSID.

DSLMRCQR Control Queue
The queue is used in the receive process RPROC1. It contains the received MQI reply and report messages which are not yet finally processed. The queue must be defined with KEY1 and KEY2 in the DSLFNTT. KEY1 represents the receive process name concatenated with the receive queue index. It is contained in the TOF field DSLKKEY1. KEY2 represents the *MsgId* field of the MQI control block MQMD. It is contained in the TOF field DSLKMSID.

The routing table (here, the sample routing table DSLKQRT) must be associated to the control queue.

DSLMRCQ1 Control Queue
The queue is used in the receive process RPROC2. It contains the received MQI request and, for MVS, report messages which are not yet finally processed. The queue must be defined with KEY1 and

KEY2 in the DSLFNTT. KEY1 represents the receive process name concatenated with the receive queue index. It is contained in the TOF field DSLKKEY1. KEY2 represents the *MsgId* field of the MQI control block MQMD. It is contained in the TOF field DSLKMSID.

The routing table (here, the sample routing table DSLKQRT) must be associated to the control queue.

DSLMRCQS **Control Queue**
The queue is used in the send processes SPROC1 and SPROC2. It contains the sent MQI request and reply messages which are not yet finally processed. The queue must be defined with KEY1 and KEY2 in the DSLFNTT. KEY1 represents the send process name concatenated with the MERVA ESA send queue name. It is contained in the TOF field DSLKKEY1. KEY2 represents the *MsgId* field of the MQI control block MQMD. It is contained in the TOF field DSLKMSID.

The routing table (here, the sample routing table DSLKQRT) must be associated to the control queue.

DSLMRDE0 **Data Entry Queue**
The queue is used to create SWIFT messages for the scenario.

DSLMRDMY **Dummy Queue**
The queue is used to get rid of obsolete messages. In the scenario, this applies to:

- Reply messages after the messages were sent
- Reply and report messages after the messages were correlated

DSLMRERR **Recovery Error Queue**
The queue is used to store the messages after an error occurred during message recovery. It is also used to store unsupported MQI report messages.

DSLMRNCO **Correlation Error Queue**
The queue is used to store MQI report and reply messages which could not be correlated.

DSLMRRQ1 **Received Messages Queue**
The queue is used to store the received or correlated messages which are finally processed. Optionally, the queue can be defined with KEY1 in the DSLFNTT. In the scenario, KEY1 represents the status of the messages. It is contained in the TOF field DSLKSTAT.

DSLMRSNT **Sent Messages Queue**
The queue is used to store the sent messages which are finally processed. This applies only to MQI datagrams which are not used in the scenario.

DSLMRSQ1 **Send Queue**
The queue is used in the send process SPROC1. It contains the MERVA ESA messages which are to be sent as MQI request messages. For an automatic start, the transaction code DSLS or any other valid transaction code of the MERVA-to-MQI send process transaction must be associated to the send queue.

DSLMRSQ2 **Send Queue**
The queue is used in the send process SPROC2. It contains the received MQI request messages which are to be prepared and sent as MQI reply messages. For an automatic start, the transaction

code DSLS or any other valid transaction code of the MERVA-to-MQI send process transaction must be associated to the send queue.

DSLMRSTR **Start Queue**
The queue is specified in the receive processes RPROC1 and RPROC2. It can be used to start these receive processes via the operator command SF (Start Function). The start queue must be defined as a dummy queue in the DSLFNTT. The transaction code DSLR or any other valid transaction code of the MQI-to-MERVA receive process transaction must be associated to the start queue.

DSLMRSTS **Start Queue**
The queue is specified in the send processes SPROC1 and SPROC2. It can be used to start these send processes via the operator command SF (Start Function). The start queue must be defined as a dummy queue in the DSLFNTT. The transaction code DSLS or any other valid transaction code of the MERVA-to-MQI send process transaction must be associated to the start queue.

Message Flow between MERVA ESA and MQSeries

The send and receive processes are described in "Installation Verification Scenario" on page 89. The processing steps in the scenario are:

1. A SWIFT message is put or routed to the MERVA ESA queue DSLMRSQ1. This queue is the MERVA ESA send queue in the send process SPROC1.
2. The SWIFT message in the send queue DSLMRSQ1 activates the send process SPROC1. The MERVA-MQI Attachment retrieves the SWIFT message from the send queue DSLMRSQ1, calls the user exit 8001 which adds a field to the message, and puts the modified message to the associated MQI send queue DSL.MQI.SNDRCV as an MQI request message.

   The MERVA-MQI Attachment sets report options to get MQI COA and COD report messages for the request message. A copy of the sent request message waits in the COA wait queue DSLMRCOA for the COA report.
3. When placed on the MQI send queue DSL.MQI.SNDRCV, the queue manager generates a COA report for the request message, and puts the COA report to the reply-to queue DSL.MQI.REPLY_TO_Q specified in send process SPROC1.
4. The COA report in the reply-to queue DSL.MQI.REPLY_TO_Q activates the receive process RPROC1. The MERVA-MQI Attachment retrieves the COA report from the reply-to queue DSL.MQI.REPLY_TO_Q, correlates the COA report with the request message in the COA wait queue DSLMRCOA, and puts the correlated request message to the COD wait queue DSLMRCOD.
5. The request message in the MQI send queue DSL.MQI.SNDRCV activates the receive process RPROC2. The MERVA-MQI Attachment retrieves the request message from the MQI send queue DSL.MQI.SNDRCV, calls the user exit 8002 which writes the additional field in the message to the MSGACK field, and routes the request message to the target queues DSLMRRQ1 and DSLMRSQ2. DSLMRRQ1 is the receive queue for the request message. DSLMRSQ2 is the send queue in the send process SPROC2. After commit to the MQSeries, the request message is deleted from the MQI send queue DSL.MQI.SNDRCV.
6. When deleted from the MQI send queue DSL.MQI.SNDRCV, the queue manager generates a COD report for the request message, and puts the COD report to the reply-to queue DSL.MQI.REPLY_TO_Q specified in send process SPROC1.

7. The COD report in the reply-to queue DSL.MQI.REPLY_TO_Q activates the receive process RPROC1. The MERVA-MQI Attachment retrieves the COD report from the reply-to queue DSL.MQI.REPLY_TO_Q, correlates the COD report with the request message in the COD wait queue DSLMRCOD, and routes the correlated request message to the ACK wait queue DSLMRAWQ where it expects the MQI reply message.
8. The request message in the MERVA ESA send queue DSLMRSQ2 activates the send process SPROC2. The MERVA-MQI Attachment retrieves the request message from the send queue DSLMRSQ2 and creates an MQI reply message which has no application data. Then the MERVA-MQI Attachment calls user exit 8001 which adds a field as application data to the reply message, and puts the reply message to the reply-to queue DSL.MQI.REPLY_TO_Q specified in send process SPROC1.

   For MVS, the MERVA-MQI Attachment sets report options to get MQI COA and COD report messages for the reply message. A copy of the sent reply message waits in the COA wait queue DSLMRCOA for the COA report.

   Note: In send process SPROC2, the MERVA-MQI Attachment does **not** put the reply message to the associated MQI send queue DSL.MQI.SNDRCV. When a reply message is to be sent, the MERVA-MQI Attachment always puts it to the reply-to queue and ignores the associated MQI send queue.
9. **MVS only:** When placed on the reply-to queue DSL.MQI.REPLY_TO_Q, the queue manager generates a COA report for the reply message, and puts the COA report to the reply-to queue DSL.MQI.SNDRCV specified in send process SPROC2.
10. **MVS only:** The COA report in the reply-to queue DSL.MQI.SNDRCV activates the receive process RPROC2. The MERVA-MQI Attachment retrieves the COA report from the reply-to queue DSL.MQI.SNDRCV, correlates the COA report with the reply message in the COA wait queue DSLMRCOA, and puts the correlated reply message to the COD wait queue DSLMRCOD.
11. The reply message in the reply-to queue DSL.MQI.REPLY_TO_Q activates the receive process RPROC1. The MERVA-MQI Attachment retrieves the reply message from the reply-to queue DSL.MQI.REPLY_TO_Q, correlates the reply message with the request message in the ACK wait queue DSLMRAWQ, calls the user exit 8002 which writes the reply message data to the MSGACK field, and routes the correlated request message to the target queue DSLMRRQ1. After commit to the MQSeries, the reply message is deleted from the reply-to queue DSL.MQI.REPLY_TO_Q.
12. **MVS only:** When deleted from the reply-to queue DSL.MQI.REPLY_TO_Q, the queue manager generates a COD report for the reply message, and puts the COD report to the reply-to queue DSL.MQI.SNDRCV specified in send process SPROC2.
13. **MVS only:** The COD report in the reply-to queue DSL.MQI.SNDRCV activates the receive process RPROC2. The MERVA-MQI Attachment retrieves the COD report from the reply-to queue DSL.MQI.SNDRCV, correlates the COD report with the reply message in the COD wait queue DSLMRCOD, and routes the correlated reply message to the target queue DSLMRRQ1.

Verifying and Starting a Connection to MQSeries

The MERVA-MQI Attachment requires that a connection has been established between the subsystems MQSeries and CICS or IMS.

If you are using MQSeries for VSE/ESA, you can continue with "Running the Installation Verification Scenario" on page 100.

Connection from CICS/ESA to MQSeries for MVS/ESA

You can use the CICS adapter control panels to display details of the connection. To access the adapter control panels, you must be authorized to use the transaction code CKQC:

1. After your CICS was started, type CKQC and press Enter.

   The CICS adapter control initial panel is displayed.
2. Select **Connection** from the menu bar at the top of the screen using the TAB key and press Enter.

   A pull-down menu is displayed.
3. Select the **Display** action from the pull-down menu and press Enter.

   Either the Display Connection panel or the CICS adapter messages panel is displayed.

If the Display Connection panel is displayed and the connection status shows 'Connected', the connection has been made. Then proceed as follows:

1. Press PF key F12 to leave the Display Connection panel.

   The CICS adapter control initial panel showing the pull-down menu is displayed.
2. Press PF key F12 to leave the pull-down menu.

   The CICS adapter control initial panel without the pull-down menu is displayed.
3. Press PF key F3 to leave the CICS adapter control initial panel.

Otherwise, you can try to start the connection. Do the following:

1. Press PF key F12 to leave the Display Connection panel or the CICS adapter messages panel.

   The CICS adapter control initial panel showing the pull-down menu is displayed.
2. Select the **Start** action from the pull-down menu and press Enter.

   A secondary parameter window is displayed.
3. Modify the connection values displayed in the **Start a Connection** secondary parameter window. Alternatively, use the defaults derived from the INITPARM or EXEC PARM settings, if defined.
4. Press Enter to confirm.

   The CICS adapter messages panel is displayed. The panel shows messages indicating the success or failure of the attempt to start the connection.
5. Press PF key F12 to leave the CICS adapter messages panel.

   The CICS adapter control initial panel showing the pull-down menu is displayed.
6. Press PF key F12 to leave the pull-down menu.

   The CICS adapter control initial panel without the pull-down menu is displayed.
7. Press PF key F3 to leave the CICS adapter control initial panel.

Note: When CICS and MQSeries are connected, you should also verify whether an instance of the MQSeries-supplied task initiator CKTI was started. The task initiator is the MQSeries trigger monitor.

Enter CKQC and select **CKTI** from the menu bar at the top of the CICS adapter control initial panel using the TAB key. Follow the instructions on the pull-down menu and the secondary window to display details of the current instances of CKTI. If required, start an instance of CKTI selecting the **Start** action from the pull-down menu.

Connection from IMS/ESA to MQSeries for MVS/ESA

You can use IMS operator commands to control and monitor the connection. You must be authorized to issue these IMS commands:

- /DISPLAY SUBSYS *sysid*
- /START SUBSYS *sysid*

where *sysid* is the MQSeries subsystem name, for example, CSQ1.

/DISPLAY SUBSYS *sysid* displays connection status and thread activity.
/START SUBSYS *sysid* connects the IMS control region to an MQSeries subsystem.

In an MPP region, IMS makes a thread connection when the first application program is scheduled into that region. The thread is retained for the duration of the region or until the connection is stopped.

When you issue the command /DISPLAY SUBSYS *sysid*, IMS shows which program is active on each dependent region (MPP or BMP region), the LTERM name, and the control region connection status. When a connection between IMS and MQSeries exists, the status for the MQSeries subsystem shows CONN or CONNECTED. The thread status from a dependent region is shown as one of the following:

- CONN
- CONN, ACTIVE (including LTERM)

If the status for the MQSeries subsystem indicates that there is no connection between IMS and MQSeries, you can try to start the connection. Issue the command /START SUBSYS *sysid* to start the connection.

The command can be issued regardless of whether MQSeries is active or not, and does not imply that the connection has been established. You must issue the /DISPLAY SUBSYS *sysid* command to see whether the connection is available.

Note: In addition to connecting the IMS control region to MQSeries, a connection from the MPP region where the MERVA-MQI Attachment send or receive transaction runs, to MQSeries must also be established. IMS initializes and creates this connection automatically doing the following:

1. It reads the subsystem member (SSM) from IMS.PROCLIB.
2. It loads the MQSeries adapter.

Running the Installation Verification Scenario

In order to verify the installation of the MERVA-MQI Attachment, do the following:

1. **For MVS**, execute the SMP/E USERMOD job DSLKPEYA, which is part of the installation library SDSLINS0.

   For VSE, assemble the source member DSLKPSMV, then run the linkbook LSLKPSMV. Both the source member and the linkbook are parts of the sublibrary MERVA.LIBS.

2. Start your MERVA ESA system.

   For MVS, if you are running your MERVA ESA system on an IMS installation, run each of the following in its own MPP region:

   - The MERVA ESA end user driver DSLEUD
   - The MERVA-MQI Attachment transaction program DSLKQR
   - The MERVA-MQI Attachment transaction program DSLKQS

3. **For MVS**, ensure that a connection to MQSeries exists. Proceed as described in “Connection from CICS/ESA to MQSeries for MVS/ESA” on page 99 or “Connection from IMS/ESA to MQSeries for MVS/ESA” on page 100.

   For VSE, ensure that MQSeries is operable. To do this, you must be authorized to use the transaction code MQSE. After CICS is started, enter **MQSE I**. If MQSeries is operable, the response will be:

   ```
   MQSE: MQSeries environment setup and initialized
   ```

4. Sign on to MERVA ESA as a MERVA ESA master user.
5. Create the new MERVA ESA user MQIUSER via the user file maintenance function.

   Specify the user type K to indicate that MQIUSER is a MERVA-MQI Attachment user.

   Specify the appropriate SWIFT address (BIC) as the origin ID.

   Assign the following functions to MQIUSER:

   - CMD
   - MSC
   - DSLMR*

   With DSLMR* the queues described in “MERVA ESA Queues” on page 95 are assigned as functions.

   Assign at least message type S100 to MQIUSER.

6. Sign off from MERVA ESA.
7. Sign on to MERVA ESA with the new user ID MQIUSER.
8. Select data entry queue DSLMRDE0 from the Function Selection panel.
9. Enter MT S100 in the command line and press Enter.
10. Enter the data for a SWIFT input message and press PF key F3 (EOM). The message moves to the MERVA-MQI Attachment send queue DSLMRSQ1 and the send process SPROC1 is initiated.

You can monitor the activities of the send and receive processes when you inspect the operator messages issued by the MERVA-MQI Attachment:

1. Press PF key F3 (Return) to return to the Function Selection panel.
2. Select function CMD or MSC from the Function Selection panel.
3. Enter DM DSL6 in the command line and press Enter.
4. If required, press PF key F2 (Repeat) until all operator messages have been displayed.

In order to see the results of the installation verification test, do the following:

1. Press PF key F3 (Return) to return to the Function Selection panel.
2. Select function DSLMRRQ1 from the Function Selection panel.
3. In function DSLMRRQ1, press PF key F11 (List).

   For MVS: Three messages are expected to be in the list. Each message must have a unique status:

- ACK
- COD
- RCVD

For VSE: Two messages are expected to be in the list. Each message must have a unique status:

- ACK
- RCVD

4. Select the message with the status RCVD and press PF key F4 (Get QSN).

   This is the received MQI request message. The following message must be shown:

   ```
   This is additional data from the user exit DSLKQ001
   ```

   Enter SHOW KCOV in the command line and press Enter. You see the first page of the MQI control block MQMD. Press PF key F8 (Page +1) to page forward through the MQI control block data until you see the header data of the SWIFT message. Now press Enter to return to the display without the MQI control block data.
5. Press PF key F12 (Escape) to return to the list.
6. Select the message with the status ACK and press PF key F4 (Get QSN).

   This is the MQI request message correlated with an MQI reply message. The following message must be shown:

   ```
   This is a reply from the user exit DSLKQ001 of send process SPROC2
   ```

   Enter SHOW KCOV in the command line and press Enter. You see the first page of the MQI control block MQMD. Press PF key F8 (Page +1) to page forward through the MQI control block data until you see the header data of the SWIFT message. Now press Enter to return to the display without the MQI control block data.
7. Press PF key F12 (Escape) to return to the list.
8. **MVS only:** Select the message with the status COD and press PF key F4 (Get QSN).

   This is the MQI reply message correlated with an MQI COD report message.

   Enter SHOW KCOV in the command line and press Enter. You see the first page of the MQI control block MQMD. The field Feedback contains decimal 260 and hexadecimal 104. This feedback code indicates that the MQSeries queue manager sent a Confirm-on-delivery (COD) report (see the *MQSeries Application Programming Reference* for details). Press PF key F8 (Page +1) to page forward through the MQI control block data until you see the reply message data. Now press Enter to return to the display without the MQI control block data.
9. **MVS only:** Press PF key F12 (Escape) to return to the list.
10. Press PF key F3 (Return) to return to the Function Selection panel.
11. Press PF key F3 (Signoff) to sign off from MERVA ESA.

Chapter 17. Verifying the Installation of FMT/ESA Using MERVA-MQI Attachment

Before you can verify that FMT/ESA using MERVA-MQI Attachment was correctly installed, you must first verify the installation of MERVA-MQI Attachment as described in “Chapter 16. Verifying the Installation of MERVA-MQI Attachment” on page 89.

To verify FMT/ESA using MERVA-MQI Attachment, four processes (two send processes and two receive processes) are used. These processes are defined in the sample process tables DSLKPFMT (for MVS) and DSLKPFMV (for VSE), and are shown in the following figures. All MQI and MERVA queues affected by these processes, including the ISN and OSN control queues (DSLISNCQ and DSLOSNCQ), were defined in the queue definition steps in Chapter 16.

Sample Definitions

```
DSLKPROC TYPE=SEND,                                           *
      NAME=SPROC1,                                            *
      ALLSNDQ=((DSLMRSQ1,DSL.MQI.SNDRCV)),                    *
      COAWQ=DSLMRCOA,          SWIFT ACK LOCALLY GENERATED    *
      CODWQ=#,                 DELIVERY NOTIFICATION (MT 011) *
      MQICTLQ=DSL.MQI.CONTROL,                                *
      MRVCTLQ=(DSLMRCQS,CONTINUE),                            *
      MRVSTAQ=DSLMRSTS,                                       *
      REPLYTQ=DSL.MQI.REPLY_TO_Q,                             *
      EXIT=8044,                                              *
      ISNCTLQ=DSLISNCQ,        REQUIRED FOR EXIT=8044         *
      OPMSDM=SUBSET
```

Figure 31. Send Process SPROC1

```
DSLKPROC TYPE=SEND,                                           *
      NAME=SPROC2,                                            *
      ALLSNDQ=((DSLMRSQ2,DSL.MQI.SNDRCV)),                    *
      MQICTLQ=DSL.MQI.CONTROL,                                *
      MRVCTLQ=(DSLMRCQS,CONTINUE),                            *
      MRVSTAQ=DSLMRSTS,                                       *
      EXIT=8044,                                              *
      ISNCTLQ=DSLISNCQ,        REQUIRED FOR EXIT=8044         *
      OPMSDM=SUBSET
```

Figure 32. Send Process SPROC2 for MVS

```
DSLKPROC TYPE=SEND,                                            *
      NAME=SPROC2,                                             *
      ALLSNDQ=((DSLMRSQ2,DSL.MQI.SNDRCV)),                     *
      MQICTLQ=DSL.MQI.CQ,                                      *
      MRVCTLQ=(DSLMRCQS,CONTINUE),                             *
      MRVSTAQ=DSLMRSTS,                                        *
      EXIT=8044,                                               *
      ISNCTLQ=DSLISNCQ,        REQUIRED FOR EXIT=8044          *
      OPMSDM=SUBSET
```

Figure 33. Send Process SPROC2 for VSE

```
DSLKPROC TYPE=RECEIVE,                                         *
      NAME=RPROC1,                                             *
      MQIRCVQ=(DSL.MQI.REPLY_TO_Q),                            *
      MRVCTLQ=DSLMRCQR,                                        *
      MRVSTAQ=DSLMRSTR,                                        *
      EXIT=8044,                                               *
      OSNCTLQ=DSLOSNCQ,        REQUIRED FOR EXIT=8044          *
      ISNCTLQ=DSLISNCQ,        REQUIRED FOR EXIT=8044          *
      OPMSDM=SUBSET
```

Figure 34. Receive Process RPROC1

```
DSLKPROC TYPE=RECEIVE,                                         *
      NAME=RPROC2,                                             *
      MQIRCVQ=(DSL.MQI.SNDRCV),                                *
      MRVCTLQ=DSLMRCQ1,                                        *
      MRVSTAQ=DSLMRSTR,                                        *
      EXIT=8044,                                               *
      OSNCTLQ=DSLOSNCQ,        REQUIRED FOR EXIT=8044          *
      ISNCTLQ=DSLISNCQ,        REQUIRED FOR EXIT=8044          *
      OPMSDM=SUBSET
```

Figure 35. Receive Process RPROC2

Running the FMT/ESA Installation Verification Scenario

In order to verify the installation of FMT/ESA using MERVA-MQI Attachment, do the following:

1. **For MVS**, execute the SMP/E USERMOD job DSLKPEYB, which is part of the installation library SDSLINS0.

   For VSE, assemble the source member DSLKPFMV, then run the linkbook LSLKPFMV. Both the source member and the linkbook are parts of the sublibrary MERVA.LIBS.
2. Start your MERVA ESA system.

   For MVS, if you are running your MERVA ESA system on an IMS installation, run each of the following in its own MPP region:
   - The MERVA ESA end user driver DSLEUD
   - The MERVA-MQI Attachment transaction program DSLKQR
   - The MERVA-MQI Attachment transaction program DSLKQS
3. **For MVS**, ensure that a connection to MQSeries exists. Proceed as described in “Connection from CICS/ESA to MQSeries for MVS/ESA” on page 99 or “Connection from IMS/ESA to MQSeries for MVS/ESA” on page 100.

For VSE, ensure that MQSeries is operable. To do this, you must be authorized to use the transaction code MQSE. After CICS is started, enter **MQSE I**. If MQSeries is operable, the response will be:

```
MQSE: MQSeries environment setup and initialized
```

4. Sign on to MERVA ESA with the user ID MQIUSER created during the MERVA-MQI Attachment installation verification test.
5. Select the user file maintenance function and add the following functions to MQIUSER, which has already been assigned the functions CMD, MSC, and DSLMR*:
   - AUT
   - DSLISNCQ
   - DSLOSNCQ
6. Press PF key F3 (Return) to return to the Function Selection panel.
7. Select the authenticator-key file maintenance function AUT and create both a sending and a receiving authenticator key.
8. Select function CMD from the Function Selection panel.
9. Press PF key F10 (DP) to display the status of MERVA ESA programs.
10. Check whether the SWIFT authentication initialization program DWSAUTIN was started. Look for the descriptive name of DWSAUTIN, for example SWIFTAUT. If its status is not ACTIVE, enter in the command line **start xx**, where **xx** can be either the descriptive name or the program identification shown in column PID.
11. Press PF key F3 (Return) to return to the Function Selection panel.
12. Select data entry queue DSLMRDE0 from the Function Selection panel.
13. Enter MT S100 in the command line and press Enter.
14. Enter the data for a SWIFT input message. Enter priority **U** and delivery monitoring **3** into the appropriate application header fields. Press PF key F3 (EOM). The message moves to the MERVA-MQI Attachment send queue (DSLMRSQ1), and the send process SPROC1 is initiated.

To monitor the activities of the send and receive processes, when you inspect the operator messages issued by the MERVA-MQI Attachment:

1. Press PF key F3 (Return) to return to the Function Selection panel.
2. Select function CMD or MSC from the Function Selection panel.
3. Enter DM DSL6 in the command line and press Enter.
4. If required, press PF key F2 (Repeat) until all operator messages have been displayed.

In order to see the results of the installation verification test:

1. Press PF key F3 (Return) to return to the Function Selection panel.
2. Select function DSLMRRQ1 from the Function Selection panel.
3. In function DSLMRRQ1, press PF key F11 (List). There should be three messages in the list: one with status COA, and two with status RCVD.
4. Select the message with the status COA and press PF key F4 (Get QSN). This is the SWIFT input message containing the SWIFT acknowledgment and, in the basic header, the session number 0000 and the ISN 000001. Press PF key F8 (Page +1) to page forward until you see the trailer field, which contains the MAC trailer.
5. Enter **SHOW KCOV** in the command line and press Enter. You see the first page of the MQI control block MQMD. The Feedback field contains decimal

259 and hexadecimal 103. This feedback code indicates that the MQSeries queue manager sent a confirm-on-arrival (COA) report due to the value of the COAWQ parameter of send process SPROC1. Press PF key F8 (Page +1) to display the second page of the MQMD. The field ApplIdentityData contains the value **W** as the first character, and **1** as the next to the last character. The W indicates that the message is a SWIFT message. The value 1 results from the parameter specification **CODWQ=#** in the send process SPROC1, and indicates that a delivery notification is to be generated if the message requests this in its application header. This is the case for the current message.

6. Press Enter to return to the display without the MQI control block data.
7. Press PF key F12 (Escape) to return to the list.
8. Select the first message with the status RCVD and press PF key F4 (Get QSN). If it is a delivery notification, press PF key F12 (Escape) to return to the list, select the second message with the status RCVD and press PF key F4 (Get QSN).
9. The current message is the SWIFT output message generated from the received SWIFT input message. It should contain the MERVA message DWS765I indicating a successful authentication. The basic header should contain the session number 0000 and the OSN 000001. Press PF key F8 (Page +1) to page forward until you see the trailer field, which should contain the MAC trailer.
10. Press PF key F12 (Escape) to return to the list.
11. Select the other message with the status RCVD and press PF key F4 (Get QSN). The current message is the delivery notification generated from the received SWIFT input message. The basic header should contain the session number 0000 and the OSN 000002.
12. Press PF key F12 (Escape) to return to the list.
13. Press PF key F3 (Return) to return to the Function Selection panel.
14. Select function DSLISNCQ from the Function Selection panel.
15. In function DSLISNCQ, press PF key F11 (List).
16. Select the only message and press PF key F4 (Get QSN). The ISN 000001 for send process SPROC1 should be displayed.
17. Press PF key F12 (Escape) to return to the list.
18. Press PF key F3 (Return) to return to the Function Selection panel.
19. Select function DSLOSNCQ from the Function Selection panel.
20. In function DSLOSNCQ, press PF key F11 (List).
21. Select the only message and press PF key F4 (Get QSN). The OSN 000002 should be displayed.
22. Press PF key F12 (Escape) to return to the list.
23. Press PF key F3 (Return) to return to the Function Selection panel.
24. Press PF key F3 (Signoff) to sign off from MERVA ESA.

Chapter 18. Verifying the Installation of MERVA Link USS

After installation of the MERVA ESA functions executing in the OS/390 UNIX System Services (USS) environment you may wish to perform a short test of basic MERVA Link USS functions. This chapter describes the MERVA Link USS installation verification procedure.

When you have successfully executed the installation verification procedure, you may wish to get more experience with MERVA Link USS. A short exploration of MERVA Link USS functions is described.

Installation Verification Procedure

A sample environment and configuration is defined by MERVA Link USS, and is used for the MERVA Link USS installation verification.

The MERVA Link USS Installation Verification can be executed in any OS/390 USS shell environment, for example in the USS shell provided by OS/390 in the TSO/E environment, or in a remote login shell opened at an AIX® host.

Start the MERVA Link USS Daemon (ACD)

Enter **scd** to start the MERVA Link USS Application Control Daemon (ACD) with its built-in sample configuration. The command **scd** was defined as an alias for the command **$DSLPP_DIR/cmd/ekascd.cmd** when the MERVA USS instance was generated. The content of the command shell script **ekascd.cmd** is

```
${DSLPP_DIR}/bin/ekaacd ${MERVA_DIR} trc cfg sample &
```

Provide Security Information (ACS)

Enter **acs** to call the MERVA Link USS conversation security application ACS and provide the following information when you are prompted:

- The partner MERVA Link node name. Use the name **ussnode1**.
- The client's user name, for example **merva1**.
- The client's password.

The command **acs** was defined as an alias for the command **$DSLPP_DIR/bin/ekaacs** when the MERVA USS instance was generated. It calls the MERVA Link USS program **ekaacs** with its default parameters.

You can specify the name and the password of any OS/390 USS user who is authorized to access the MERVA Link USS server resources.

This step creates the MERVA Link USS security file **msg.USSNODE1** in the **/sec** subdirectory of the MERVA USS instance directory. This file contains conversation security information in encrypted format to access the MERVA Link Messaging Server in partner node **USSNODE1**, which is the same as our local node.

Start the MERVA Link USS Control Application (ACC)

Enter **acc** to start the MERVA Link USS Application Control Command Application (ACC) in conversation mode. From now on, all shell input is interpreted as an ACC command. Enter **x** or **end** to terminate the interactive ACC process.

The command **acc** has been defined as an alias for the command **$DSLPP_DIR/bin/ekaacc sc** when the MERVA USS instance was generated.

Send Probe to the Local ASP (BTB)

Enter ACC command **pta asp1** to send a MERVA Link Probe from the local ASP ASP1 to itself (Back-to-Back). The expected command protocol (input and response) is as follows:

```
ekaacc pta asp1
EKAACC pta asp1

EKAACC_pta Send T-Probe requested
EKAACC prb Probe is sent to partner ASP ASP1 at USSNODE1

EKAACC_prb Probe was confirmed
```

The MERVA Link USS installation has been successfully verified if you get this response. If the Probe is not confirmed, you can repeat this test and ask for a processing trace using the ACC command **pta2 asp1**. The ACC command output including the trace is again written to your terminal.

If the Probe is not confirmed but the receiving MERVA Link process was successfully startet by InetD, you may find error diagnostic information in the trace file **/tmp/ekatci.trace** or in an ekatci trace file in the directory **/u/merva1/trc**.

Look at the Inbound Processing Trace

The previous step of the installation verification was supposed to write a MERVA Link USS processing trace to the **trc** subdirectory of the MERVA USS instance directory. The trace file name starts with **ekatci.t.** and is followed by a date/time stamp and an index.

Verify that the trace has been written, and have a look into the trace using a text editor. If you use the two alternatives to start the MERVA Link USS messaging server, you may wish to compare the two trace files and note the difference.

Exploring the MERVA Link USS Sample Node

After executing the installation verification, you may wish to explore more ACC functions and make preparations for your own MERVA Link USS configuration. For a more detailed description of the MERVA Link USS control facilities used in this section, refer to *MERVA for ESA Operations Guide* and *MERVA for ESA Advanced MERVA Link*.

Familiarize Yourself with ACC Help Information

Enter ACC command **h** to display a summary of the ACC commands. It shows that there are 11 other help commands. You may wish to enter all these commands (except command hx) to familiarize yourself with the available ACC help information.

The response to the ACC command **hx** consists of more than 100 lines. It makes no sense to issue this command in ACC conversation mode. To see the output of this help command in window pages enter the following command in the USS shell:

```
$DSLPP_DIR/bin/ekaacc hx | pg
```

The command

```
$DSLPP_DIR/bin/ekaacc hx > hx.out
```

writes the **hx** command output to the file **hx.out** in the current directory. Use a text editor to read the file **hx.out**.

Export the Sample Configuration

Enter ACC command **cxf** to export the MERVA Link USS sample configuration to the file **$MERVA_DIR/cfg/ekaact.cfg**. This file can be used as a skeleton for your own MERVA Link USS configuration. It contains definition lines for all configuration parameters supported by MERVA Link USS. In particular, you find the keywords for all configuration parameters in this configuration export file. Note, however, that ASP parameters are supported for test purposes only. Full ASP functionality is currently not available from MERVA Link USS.

For more information about the MERVA Link USS Configuration File, refer to *MERVA for ESA Customization Guide*.

Modify the MERVA Link USS Configuration

The output of the **cxf** command is contained in the file **/u/merva1/cfg/ekaact.cfg**. Use any text editor to modify this file as follows:

- Change **local_node** from USSNODE1 to MYNODE1 in the ACTH entry.
- Change **tci_trace_wrap_limit** from 0 to 5 in the ACTH entry.
- Change **partner_node** from USSNODE1 to MYNODE1 in the ACTA and ACTC entries.

Enter **vcd** to start the MERVA Link USS Configuration Verification Control Daemon (VCD) with the configuration contained in **$MERVA_DIR/cfg/ekaact.cfg**. The command **vcd** has been defined as an alias for the command **$DSLPP_DIR/cmd/ekavcd.cmd** when the MERVA USS instance was generated. The content of the command shell script **ekavcd.cmd** is

```
${DSLPP_DIR}/bin/ekavcd     ${MERVA_DIR}                \
                        cfg ${MERVA_DIR}/cfg/ekaact.cfg  \
                        rep ${MERVA_DIR}/trc/         k  &
```

Enter **vcc** to start the MERVA Link USS Configuration Verification Control Application (VCC) that is related to the VCD. The command **vcc** has been defined as an alias for the command **$DSLPP_DIR/bin/ekavcc sc** when the MERVA USS instance was generated.

Enter the VCC command **dph** to display parameters in the MERVA Link USS ACT header, and verify that your modifications are active.

To end the VCD, enter VCC command **trm daemon**.

Activate the Modified MERVA Link USS Configuration

When you are sure that the modified configuration is correct, you can copy the modified test configuration file **/u/merva1/cfg/ekaact.cfg** to the sample production configuration file **/u/merva1/cfg/ekaacd.cfg**. Use the **cp** command for this purpose.

Enter **acd** to start the MERVA Link USS Application Control Daemon (ACD) with the configuration contained in **$MERVA_DIR/cfg/ekaacd.cfg**. The command **acd** has been defined as an alias for the command $DSLPP_DIR/cmd/ekaacd.cmd when the MERVA USS instance was generated. The content of the command shell script **ekaacd.cmd** is

```
$DSLPP_DIR/bin/ekaacd     ${MERVA_DIR}                \
                      cfg ${MERVA_DIR}/cfg/ekaacd.cfg  \
                      trc                              &
```

Enter **acc** to start the MERVA Link USS Application Control Application (ACC) related to the ACD with the configuration contained in **$MERVA_DIR/cfg/ekaacd.cfg**.

Enter the ACC command **dph** to display parameters in the MERVA Link USS ACT header, and verify that your modifications are active. You can see from the modified ACT IPC resource name **ekaact.b** that an ACD process switch has been performed when the **acd** command was entered. The (old) ACD that was started by the **scd** command was automatically terminated by the new ACD.

Provide security information for the new partner node ’MYNODE1’ and enter the ACC command **pta asp1** again to test the Probe within your new MERVA Link USS node. The test should show the same result as the Probe test of the installation verification. However, the trace file name is different because a wrap limit has been specified. The MERVA Link USS inbound TCP/IP trace files start with **ekatci.trace** when a wrap limit is specified.

To end the ACD, enter ACC command **trm daemon**.

Part 4. Maintaining MERVA ESA and Its Components

Chapter 19. Maintaining MERVA ESA

After you have installed MERVA ESA you may wish to change the system. If you used SMP/E under MVS to install the system, you should use SMP/E to maintain or customize the system. If you used MSHP under VSE to install the system, you should use MSHP to maintain or customize the system.

Details about making such changes are described in the following books:

- For MVS SMP/E, in the *System Modification Program Extended Reference*
- For VSE/AF MSHP, in the *VSE/ESA System Control Statements*

Contact your IBM Marketing Representative or System Engineer (SE) for specific information about available program services. Ask the IBM Preventive Service Planning (PSP) facility to check for the latest additional information and procedures available for the MERVA ESA.

Report any difficulties you have using MERVA ESA to your IBM Support Center. If an APAR/PTF is required, refer to the *MERVA for ESA Diagnosis Guide* and submit the APAR/PTF to the location identified in the *Programming System General Information Manual* (PSGIM) F229-2228, as being responsible for the failing component(s).

Types of Fixes

Temporary corrective fix

A temporary corrective fix usually corrects source or object code members in the MERVA ESA libraries by means of an APAR fix.

For SMP/E, use the ++USERMOD statement to install the modifications. The SMP/E control data set (CDS) then shows your modification, and you are informed by SMP/E when a ++APAR or ++PTF statement overwrites your ++USERMOD.

For MSHP, use the CORRECT statement to install the modifications. The MSHP history file shows your modification, and you are informed by MSHP when a PTF overwrites your CORRECT.

Program temporary fix (PTF)

A program temporary fix (PTF) usually replaces source or object code members in the MERVA ESA libraries with a PTF tape. A PTF can supersede one or several APAR fixes or PTFs that you have already installed.

In MVS, SMP/E knows which APARs have been superseded and provides a list, which also includes the overwritten USERMODs.

In VSE, MSHP will list all modifications that are overwritten by the PTF, if these modifications have been installed with MSHP CORRECT control statements.

Maintaining with SMP/E (MVS only)

To unload a service PTF ULnnnnn, use the following SMP/E sample job:

```
//RECVAPPL JOB ACCOUNT,........                          /* MODIFY */
//*********************************************************************
//*                                                                  *
//*       THIS JOB INSTALLS PTFS IN MERVA ESA                        *
//*                                                                  *
//*       (SMP/E RECEIVE AND APPLY)                                  *
//*                                                                  *
//*********************************************************************
//*
//ULXXXXX  EXEC PGM=GIMSMP,REGION=4M,                    /* MODIFY */
//             PARM='CSI=MERVA.SMPCSI.CSI'
//*
//SMPPTFIN  DD DSN=PTF,DISP=OLD,VOL=SER=volser, /* MODIFY */
//             LABEL=(1,SL),UNIT=TAPE                    /* MODIFY */
//SMPHOLD   DD *
++NULL.
//SMPCNTL   DD *
   SET BOUNDARY (GLOBAL) .
   RECEIVE .
   SET BOUNDARY(DSLTGT) OPTIONS (DSLOPTN) .
   APPLY S (UL.....)  C(ALL) .                           /* MODIFY */
/*
```

The statements marked by /* MODIFY */ must be adapted to your installation:

- The PARM in the EXEC statement refers to the MERVA ESA SMP/E CSI to be used.
- In the SMPPTFIN DD statement define the PTF tape supplied by IBM.
- In the APPLY control statement, the PTF numbers UL..... are listed.
- To avoid syntax errors, remove the /* MODIFY */ comments before executing the job.

Note: If a USERMOD is used to modify a macro or copy code, macro replacement (++MAC SMP/E CONTROL STATEMENT IN THE USERMOD) must be used.

Maintaining with MSHP (VSE/CICS only)

Unload Service PTFs

To unload a service PTF ULnnnnn, use the following MSHP sample job:

```
// JOB PTF$UNLD ULnnnnn                              /* MODIFY */
*
// PAUSE TAPE UNIT READY ???
*
// DLBL MERVA,'MERVA.DVSE.LIBRARY',99/365,SD         /* MODIFY */
// EXTENT ,DMN002                                    /* MODIFY */
LIBDEF *,SEARCH=(MERVA.LIBS),                        /* MODIFY */  X
               CATALOG=MERVA.LIBS                    /* MODIFY */
*
// ASSGN SYS021,DISK,VOL=DMN002,SHR                  /* MODIFY */
// DLBL IJSYSHF,'MERVA.PRODUCT.HISTORY',99/365,SD    /* MODIFY */
// EXTENT SYS021,DMN002,1,0,15,15                    /* MODIFY */
*
// ASSGN SYS006,cuu,dd                               /* MODIFY */
// MTC REW,SYS006
*
// OPTION CATAL,NODUMP
// EXEC MSHP,SIZE=1024K
   INSTALL SERVICE TAPES=1
   INCLUDE PTF=(ULnnnnn)                             /* MODIFY */
/*
/&
```

To avoid syntax errors, remove the /* MODIFY */ comments before executing the job.

Connect Service PTFs

To generate installation JCL automatically, use the installation macros. The installation macros are especially useful for the installation of summary PTFs. These PTFs contain a special member called JCL generator (stage 1), which has to be modified according to the requirements of your currently used system configuration. After the modified JCL generator is assembled, the complete installation JCL is punched (stage 2). Submit the generated jobs.

The JCL generator, or a description of how to connect a single or a summary PTF, is part of the cover letter. The content of the cover letter can be listed with the following MSHP function control statement:

```
LIST  SERVICE COVER CONT
PTF=(ULnnnnn)
```

Installation Macros

DSLDPTFP
Definitions of the global JCL cards, including POWER extensions, if required.

Prerequisite to the macros DSLDPTFA, DSLDPTFL, or both.

DSLDPTFA
Punches the JCL to assemble source members, or to preedit macros.

Requires the macro DSLDPTFP as prerequisite.

DSLDPTFL
Punches the JCL for link-editing of modules.

Requires the macro DSLDPTFP as prerequisite.

The installation macros are usually called in this sequence.

Definition of the Installation Macros

The DSLDPTFP Macro

The DSLDPTFP macro serves the following:

- Defines the global JOB cards.
- Defines the global JCL cards.
- Is a prerequisite to the macros DSLDPTFA, DSLDPTFL, or both.

Name	Operator	Operands
[label]	**DSLDPTFP**	[**PWRJOB**='string ...'] [,**PWRLST**='string ...'] [,**PWRPUN**='string ...'] [,**JOBCARD**='string ...'] ,**LIBDLBL**='string ...' ,**LIBEXT**='string ...' ,**LIBDEF1**='string ...' [,**LIBDEF2**='string ...'] [,**LIBDEF3**='string ...'] ,**LIBDEFC**='string ...' [,**ASSPROA**={**DSLASSA**|*name*}] [,**ASSPROR**={**DSLASSR**|*name*}]

Notes:

label Specifies a unique statement label according to assembler language conventions.

PWRJOB
A character string that specifies the operands that must follow the operation:

```
'* $$ JOB '
```

If not specified, no POWER JOB card can be punched.

PWRLST
A character string that specifies the operands that must follow the operation:

```
'* $$ LST '
```

If not specified, no POWER LST card can be punched.

PWRPUN
A character string that specifies the operands that must follow the operation:

```
'* $$ PUN '
```

If not specified, no POWER PUN card can be punched.

JOBCARD

A character string that specifies the operands that must follow the operation:

```
'// JOB '
```

If not specified, no VSE JOB card can be punched.

LIBDLBL

A character string that specifies the operands that must follow the operation:

```
'// DLBL '
```

Note: Because of the assembler conventions, you must use double apostrophes for the operand ...,"file-id",...

LIBEXT

A character string that specifies the operands that must follow the operation:

```
'// EXTENT '
```

LIBDEF1

A character string that specifies the first sublibrary that must follow the operation, operand, or both:

```
'LIBDEF *,SEARCH=('
```

LIBDEF2

A character string that specifies the the second sublibrary in the search order of the LIBDEF statement. If not specified, LIBDEF3 is ignored.

LIBDEF3

A character string that specifies the the third sublibrary in the search order of the LIBDEF statement. If LIBDEF2 is not specified, LIBDEF3 is ignored.

LIBDEFC

A character string that specifies the sublibrary that must follow the operand:

```
'CATALOG='
```

This is the sublibrary for the output of the linkage editor.

ASSPROA

This procedure assigns the SYSIPT/SYSPCH area for the assemblies (defined during basic tape installation). The default value is DSLASSA.

ASSPROR

This procedure resets SYSIPT/SYSPCH after the assemblies (defined during basic tape installation). The default value is DSLASSR.

The DSLDPTFA Macro

The DSLDPTFA macro serves the following:

- Assemble source members.

Name	Operator	Operands
[label]	**DSLDPTFA**	**NAMES**=(name_1,name_2,...,name_n) [,**NSTEP**={<u>3</u>|n}] [,**CR**={<u>**SXREF**</u>|**XREF**|**NOXREF**}]

Notes:

label A unique statement label according to the assembler language conventions.

NAMES
The members to be processed.

NSTEP
The number of members to be processed within one job. The default is 3.

CR The cross-reference layout of assembly listings. The default is SXREF.

The DSLDPTFL Macro

The DSLDPTFL macro serves link-edit modules.

Name	Operator	Operands
[label]	**DSLDPTFL**	**NAMES**=(name_1,name_2,...,name_n)

Notes:

label A unique statement label according to the assembler language conventions.

NAMES
The members to be processed.

Part 5. Appendixes

Appendix A. DSLGEN Stage-1 Generation Macro

The DSLGEN macro is used to generate those tables of MERVA ESA that are composed of parts supplied by its components. DSLGEN also allows for adding user definitions.

There are four types of DSLGEN macros:

TYPE=INITIAL
Specifies the start of DSLGEN processing, and which MERVA ESA components are used. This must be the first statement.

TYPE=USER
Specifies additions to the MERVA ESA tables.

TYPE=FINAL
Specifies the end of DSLGEN processing. This macro may be followed by a DSLGEN TYPE=CLINK macro.

TYPE=CLINK
VSE only. Specifies that DSLGEN is to generate link-edit jobs for CICS-dependent modules. This type can be used without other DSLGEN macroinstructions (it then must contain the same parameters as the DSLGEN TYPE=INITIAL) or after the DSLGEN TYPE=FINAL (then no other parameters must be specified).

The following sections show which parameters can be used with each type.

DSLGEN TYPE=INITIAL and TYPE=CLINK

TYPE=INIT specifies the start of DSLGEN processing and which MERVA ESA components are used. This must be the first statement. TYPE=INIT also allows to change the name of some of the tables. If a name is changed, it must also be specified in the MERVA ESA customizing parameters DSLPRM.

TYPE=CLINK is used in VSE only. It specifies that DSLGEN is to generate link-edit jobs for CICS-dependent modules. This type can be used without other DSLGEN macroinstructions (it then must contain the same parameters as the DSLGEN TYPE=INITIAL) or after the DSLGEN TYPE=FINAL (then no other parameters must be specified).

Name	Operator	Operands	
	DSLGEN	**TYPE**={**INITIAL**|**CLINK**}	
		[,**ID**=(*ID1*[,*ID2*[,...[,*IDn*]]])]	
		[,**TGTZONE**=**DSLTGT**]	MVS only
		[,**OPTN**=**DSLOPTN**]	MVS only
		[,**JOBCARD**={*c*}]	VSE only
		[,**NSTEPS**={*c*}]	VSE only
		[,**CR**={*c*}]	VSE only
		[,**LIBDEF1**={*c*}]	VSE only
		[,**LIBDEF2**={*c*}]	VSE only

Name	Operator	Operands
		[,**LIBDEF3**={*c*}] VSE only [,**LIBDEFC**={*c*}] VSE only [,**LIBDLBL**={*c*}] VSE only [,**LIBEXT**={{*c*}] VSE only [,**SUBLIB**={*c*}] VSE only [,**ASSIGNL**={*c*}] VSE only [,**ASSIGNR**={*c*}] VSE only [,**PWRJOB**={*c*}] VSE only [,**PWRLST**={*c*}] VSE only [,**PWRPUN**={*c*}] VSE only [,**EPT**={**DSLEPTT**|**NO**}] [,**FDT**={**DSLFDTT**|**NO**|*cccccccc*}] [,**FLT**={**DSLFLTT**|**NO**|*cccccccc*}] [,**FNT**={**DSLFNTT**|**NO**|*cccccccc*}] [,**MCM**={**DSLEMCMT**|**NO**}] [,**MPT**={**DSLMPTT**|{**NO**}] [,**MSG**={**DSLMSGT**|**NO**|*cccccccc*}] [,**MTT**={**DSLMTTT**|**NO**|*cccccccc*}] [,**NCM**={**DSLNCMT**|**NO**}] [,**NPT**={**DSLNPTT**|**NO**}] [,**NTR**={**DSLNTRT**|**NO**}]

Notes

ID The MERVA ESA components for which definition statements are to be generated:

SWIFT or SWIFTII
SWIFT Link

TELEX Telex Link, both via a fault-tolerant system and via workstation

MLINK MERVA Link

FMT MERVA-to-MERVA Financial Message Transfer/ESA using MERVA Link

RECON The Traffic Reconciliation feature (MVS only)

The default is ID=(SWIFT,TELEX,MLINK,FMT).

TGTZONE (MVS only)
Defines the SMP/E target zone to be used for MERVA ESA. The default is DSLTGT.

OPTN (MVS only)
Defines the SMP/E options entry to be used for MERVA ESA. The default is DSLOPTN.

JOBCARD (VSE only)
Allows the user to specify a VSE job statement. The jobname is mandatory, must be in the first position, and must not be longer than 6 characters. The default is JOBNAM (XXXX,Y),LP.

NSTEPS (VSE only)
Specifies the maximum number of steps in each assembly job. The default is 3.

CR (VSE only)
Specifies the assembly cross-reference requirements. The default is SXREF.

LIBDEF1 (VSE only)
Defines the first VSE sublibrary name to be used in the SEARCH operand of the LIBDEF statement. This parameter is mandatory (it is the first one in the search order). If not entered, it will be defaulted to the specification of the SUBLIB parameter.

LIBDEF2 (VSE only)
Defines the second VSE sublibrary name to be used in the SEARCH operand of the LIBDEF statement. This parameter is optional and there is no default value. When present, it is the second one in the search order.

LIBDEF3 (VSE only)
Defines the third VSE sublibrary name to be used in the SEARCH operand of the LIBDEF statement. This parameter is optional and there is no default value. When present, it is the third one in the search order.

LIBDEFC (VSE only)
Defines the VSE sublibrary name for the linkage editor to be used in the CATALOG operand of the LIBDEF statement. This parameter is mandatory. If not entered, it will be defaulted to the specification of the LIBDEF1 parameter.

LIBDLBL (VSE only)
Defines the DLBL statement information for the MERVA ESA production library system in VSE. This is a required parameter.

LIBEXT (VSE only)
Defines the EXTENT information for the MERVA ESA production library system in VSE. This is a required parameter.

SUBLIB (VSE only)
Defines the name of the VSE sublibrary that is used by the VSE LIBRARIAN program to catalog the text deck after an assembly. This parameter is optional. The default is MERVA.LIBS.

ASSIGNL (VSE only)
Defines the name of the procedure that is used to assign the private SYSIPT and SYSPCH data sets for the assembly steps. This parameter is optional. The default is DSLASSA.

ASSIGNR (VSE only)
Defines the name of the procedure that is used to reset the private SYSIPT and SYSPCH data set assignments to the original assignments after the assembly steps. This parameter is optional. The default is DSLASSR.

PWRJOB (VSE only)
Defines the parameters required for the POWER JOB statement. This parameter is optional. The default is JMN=INSTALL,DISP=D,CLASS=0. If you do not use POWER, VSE will treat this statement as a comment.

PWRLST (VSE only)
Defines the parameters required for the POWER LST statement. This parameter is optional. The default is DEST=(*,XXXXXXXX). If you do not use POWER, VSE will treat this statement as a comment.

PWRPUN (VSE only)
Defines the parameters required for the POWER PUN statement. This parameter is optional. The default is DEST=(*,XXXXXXXX). If you do not use POWER, VSE will treat this statement as a comment.

EPT Specifies either the name DSLEPTT for the MERVA ESA end-user driver program table (this name cannot be changed), or NO, if this generation process is not to generate DSLEPTT. If the EPT parameter is not specified, DSLEPTT is generated.

FDT Specifies either the name of the MERVA ESA field definition table or NO, if this generation process is not to generate this table. If the FDT parameter is not specified, the field definition table is generated with the name DSLFDTT.

FLT Specifies either the name of the MERVA ESA file table or NO, if this generation process is not to generate this table. If the FLT parameter is not specified, the file table is generated with the name DSLFLTT.

FNT Specifies either the name of the MERVA ESA function table or NO, if this generation process is not to generate this table. If the FNT parameter is not specified, the function table is generated with the name DSLFNTT.

MCM Specifies either the name DSLEMCMT for the MERVA ESA command table of the DSLEMSG program (this name cannot be changed), or NO, if this generation process is not to generate this table. If the MCM parameter is not specified, DSLEMCMT is generated.

MPT Specifies either the name DSLMPTT for the MERVA ESA message format service program table (this name cannot be changed), or NO, if this generation process is not to generate this table. If the MPT parameter is not specified, DSLMPTT is generated.

MSG Specifies either the name of the MERVA ESA message table or NO, if this generation process is not to generate this table. If the MSG parameter is not specified, the Message table is generated with the name DSLMSGT.

MTT Specifies either the name of the MERVA ESA message type table or NO, if this generation process is not to generate this table. If the MTT parameter is not specified, the message type table is generated with the name DSLMTTT.

NCM Specifies either the name DSLNCMT for the MERVA ESA nucleus command table (this name cannot be changed), or NO, if this generation process is not to generate this table. If the NCM parameter is not specified, DSLNCMT is generated. If DSLNCMT is generated, DSLNPTT and DSLNTRT are also generated (see NPT and NTR parameters).

NPT Specifies either the name DSLNPTT for the MERVA ESA nucleus program table (this name cannot be changed), or NO, if this generation process is not to generate this table. If the NPT parameter is not specified, DSLNPTT is generated. If DSLNPTT is generated, DSLNCMT and DSLNTRT are also generated (see NCM and NTR parameters).

NTR Specifies either the name DSLNTRT for the MERVA ESA Task Server Request table (this name cannot be changed), or NO, if this generation process is not to generate this table. If the NTR parameter is not specified, DSLNTRT is generated. If DSLNTRT is generated, DSLNCMT and DSLNPTT are also generated (see NCM and NPT parameters).

DSLGEN TYPE=USER

TYPE=USER specifies additions to the MERVA ESA tables besides the MERVA ESA components. These additions must be copy codes that are included into the table by specifying the name in the DSLGEN TYPE=USER macro.

Name	Operator	Operands
	DSLGEN	**TYPE=USER** [,**PREFIX**={**USR**|*ccc*}] [,**EPT**={**NO**|**YES**|*cccccccc*}] [,**FDT**={**NO**|**YES**|*cccccccc*}] [,**FLT**={**NO**|**YES**|*cccccccc*}] [,**FNT**={**NO**|**YES**|*cccccccc*}] [,**MCM**={**NO**|**YES**|*cccccccc*}] [,**MPT**={**NO**|**YES**|*cccccccc*}] [,**MSG**={**NO**|**YES**|*cccccccc*}] [,**MTT**={**NO**|**YES**|*cccccccc*}] [,**NCM**={**NO**|**YES**|*cccccccc*}] [,**NPT**={**NO**|**YES**|*cccccccc*}] [,**NTR**={**NO**|**YES**|*cccccccc*}]

Notes:

PREFIX
Specifies the 3-character component code (prefix) when the name of the copy code is generated if YES is specified in one of the following parameters. If the PREFIX parameter is not specified, the prefix USR is used by default. The following prefixes are reserved:

DSL For Base Functions

DWS For the SWIFT Link

ENL For the Telex Link

EKA For the MERVA Link and FMT/ESA using MERVA Link

IMR For the MERVA ESA Traffic Reconciliation

EPT Adds user code to the MERVA ESA End-User Driver Program table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and EPTTC is added to the table, for example, USREPTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

FDT Adds user code to the MERVA ESA field definition table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and FDTTC is added to the table, for example, USRFDTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

FLT Adds user code to the MERVA ESA file table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and FLTTC is added to the table, for example, USRFLTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

FNT Adds user code to the MERVA ESA function table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and FNTTC is added to the table, for example, USRFNTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

MCM Adds user code to the MERVA ESA command table of the program DSLEMSG. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and MCMTC is added to the table, for example, USRMCMTC. If a name is specified (*cccccccc*), this copy code is added to the table.

MPT Adds user code to the MERVA ESA Message Format Service program table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and MPTTC is added to the table, for example, USRMPTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

MSG Adds user code to the MERVA ESA message table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and MSGTC is added to the table, for example, USRMSGTC. If a name is specified (*cccccccc*), this copy code is added to the table.

MTT Adds user code to the MERVA ESA message type table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and MTTTC is added to the table, for example, USRMTTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

NCM Adds user code to the MERVA ESA nucleus command table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and NCMTC is added to the table, for example, USRNCMTC. If a name is specified (*cccccccc*), this copy code is added to the table.

NPT Adds user code to the MERVA ESA nucleus program table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and NPTTC is added to the table, for example, USRNPTTC. If a name is specified (*cccccccc*), this copy code is added to the table.

NTR Adds user code to the MERVA ESA task server request table. If NO is specified (which is the default), nothing is added to this table. If YES is specified, a copy code composed of the value of the PREFIX parameter and NTRTC is added to the table, for example, USRNTRTC. If a name is specified (*cccccccc*), this copy code is added to the table.

DSLGEN TYPE=FINAL

TYPE=FINAL specifies the end of DSLGEN processing. This macro can be followed by a DSLGEN TYPE=CLINK macro. All other types are ignored after TYPE=FINAL.

Name	Operator	Operands
	DSLGEN	TYPE=FINAL [,PRINT={ON|OFF}]

Additional Note:

PRINT

Specifies whether the assembly listings for MERVA ESA should be printed (PRINT=ON) or suppressed (PRINT=OFF). The default is ON. The PRINT parameter must be coded on the DSLGEN TYPE=FINAL statement.

Appendix B. MERVA ESA Table Generation Examples

VSE Table Generation Examples

Figure 36 to Figure 40 on page 130 give VSE table generation examples.

```
GENT     TITLE 'MERVA ESA VSE STAGE 1 GENERATION'
DSLGENT  DSLGEN TYPE=INITIAL,                                          +
               PWRJOB='JNM=MERTABGN,DISP=D,CLASS=0',                  +
               PWRLST='DEST=(*,LLLLLLLL)',                             +
               PWRPUN='DEST=(*,PPPPPPPP)',                             +
               JOBCARD='MERTABGN (XXXX,Y),LP',                         +
               LIBDLBL='MERVA,MERVA.DVSE.LIBRARY,99/366,SD',           +
               LIBEXT=',DMN002',                                       +
               LIBDEF1='MERVA.LIBS',                                   +
               LIBDEF2='MERVA.CIXTAB21',                               +
               LIBDEF3='MERVA.PROCS',                                  +
               LIBDEFC='MERVA.LIBS',                                   +
               SUBLIB='MERVA.LIBS'
         DSLGEN TYPE=FINAL
         END
```

Figure 36. MERVA ESA Tables Only

```
GENT     TITLE 'MERVA ESA VSE STAGE 1 GENERATION'
DSLGENT  DSLGEN TYPE=INITIAL,ID=SWIFT,                                 +
               PWRJOB='JNM=MERTABGN,DISP=D,CLASS=0',                  +
               PWRLST='DEST=(*,LLLLLLLL)',                             +
               PWRPUN='DEST=(*,PPPPPPPP)',                             +
               JOBCARD='MERTABGN (XXXX,Y),LP',                         +
               LIBDLBL='MERVA,MERVA.DVSE.LIBRARY,99/366,SD',           +
               LIBEXT=',DMN002',                                       +
               LIBDEF1='MERVA.LIBS',                                   +
               LIBDEF2='MERVA.CIXTAB21',                               +
               LIBDEF3='MERVA.PROCS',                                  +
               LIBDEFC='MERVA.LIBS',                                   +
               SUBLIB='MERVA.LIBS'
         DSLGEN TYPE=FINAL
         END
```

Figure 37. MERVA ESA and SWIFT Link Tables for the SWIFT Network

```
GENT     TITLE 'MERVA ESA VSE STAGE 1 GENERATION'
DSLGENT  DSLGEN TYPE=INITIAL,ID=SWIFT,MPT=USRMPTT,                     +
               PWRJOB='JNM=MERTABGN,DISP=D,CLASS=0',                  +
               PWRLST='DEST=(*,LLLLLLLL)',                             +
               PWRPUN='DEST=(*,PPPPPPPP)',                             +
               JOBCARD='MERTABGN (XXXX,Y),LP',                         +
               LIBDLBL='MERVA,MERVA.DVSE.LIBRARY,99/366,SD',           +
               LIBEXT=',DMN002',                                       +
               LIBDEF1='MERVA.LIBS',                                   +
               LIBDEF2='MERVA.CIXTAB21',                               +
               LIBDEF3='MERVA.PROCS',                                  +
               LIBDEFC='MERVA.LIBS',                                   +
               SUBLIB='MERVA.LIBS'
         DSLGEN TYPE=USER,MPT=YES
         DSLGEN TYPE=FINAL
         END
```

Figure 38. MERVA ESA and SWIFT Link Tables with User Additions for the SWIFT Network

Note: The DSLGEN TYPE=USER adds one copy code. With MPT=YES, the copy code USRMPTTC is added to the MERVA ESA MFS Program table USRMPTT (MPT parameter in the DSLGEN TYPE=INITIAL).

```
GENT     TITLE 'MERVA ESA VSE STAGE 1 GENERATION'
DSLGENT  DSLGEN TYPE=INITIAL,ID=SWIFT,                                 +
               PWRJOB='JNM=MERTABGN,DISP=D,CLASS=0',                  +
               PWRLST='DEST=(*,LLLLLLLL)',                             +
               PWRPUN='DEST=(*,PPPPPPPP)',                             +
               JOBCARD='MERTABGN (XXXX,Y),LP',                         +
               LIBDLBL='MERVA,MERVA.DVSE.LIBRARY,99/366,SD',           +
               LIBEXT=',DMN002',                                       +
               LIBDEF1='MERVA.LIBS',                                   +
               LIBDEF2='MERVA.CIXTAB21',                               +
               LIBDEF3='MERVA.PROCS',                                  +
               LIBDEFC='MERVA.LIBS',                                   +
               SUBLIB='MERVA.LIBS'
         DSLGEN TYPE=FINAL
         DSLGEN TYPE=CLINK
         END
```

Figure 39. MERVA ESA Tables for the SWIFT Network and Link-Edit Jobs for CICS-Dependent Modules of Base Functions (TYPE=CLINK)

```
GENT     TITLE 'CICS LINKEDITS FOR MERVA ESA VSE'
DSLGENT  DSLGEN TYPE=CLINK,                                            +
               PWRJOB='JNM=MERTABGN,DISP=D,CLASS=0',                  +
               PWRLST='DEST=(*,LLLLLLLL)',                             +
               PWRPUN='DEST=(*,PPPPPPPP)',                             +
               JOBCARD='MERTABGN (XXXX,Y),LP',                         +
               LIBDLBL='MERVA,MERVA.DVSE.LIBRARY,99/366,SD',           +
               LIBEXT=',DMN002',                                       +
               LIBDEF1='MERVA.LIBS',                                   +
               LIBDEF2='MERVA.CIXTAB21',                               +
               LIBDEF3='MERVA.PROCS',                                  +
               LIBDEFC='MERVA.LIBS',                                   +
               SUBLIB='MERVA.LIBS'
         END
```

Figure 40. Link-Edit Jobs for CICS-Dependent Modules of Base Functions

MVS Table Generation Processing Examples

```
GENT     TITLE 'MERVA ESA DSLGEN PROCESS INPUT'
DSLGENT  DSLGEN TYPE=INITIAL,ID=(SWIFT,TELEX)
         DSLGEN TYPE=FINAL
         END
```

Figure 41. MERVA ESA with the SWIFT Link and the Telex Link

In this example MERVA ESA tables are generated, which contain the definitions required for the SWIFT Link and the Telex Link. The definitions for MERVA Link are generated also, because MERVA Link is used by SWIFT Link to perform the USE functions. The DSLGEN process is part of the installation steps as described in the MERVA ESA program directories.

The installation consists of the following steps:

Unloading the distribution tape into MERVA ESA distribution libraries:

1. RECEIVE/APPLY of the product.
2. Processing JCLIN and link-edit load modules.
3. ACCEPT of the product into MERVA ESA distribution libraries.

The table generation is the last step of the installation. The updated source members for the MERVA ESA tables are generated and installed as SMP/E USERMODs. SMP/E assembles all tables and link-edits the tables DSLFDTT, DSLFLTT, DSLFNTT, DSLMSGT, and DSLMTTT. These tables do not have external references. For the tables DSLNCMT, DSLNPTT, and DSLNTRT, DSLNUC is link-edited. For the tables DSLEPTT and DSLEMCMT, DSLEUD is link-edited. For the table DSLMPTT, DSLMMFS is link-edited. At the same time the external references to modules of the SWIFT Link or the Telex Link (contained in these tables) are resolved.

Adding a User-Exit Program to the MERVA ESA Message Format Service

The following example shows how to add a user-exit program to the MERVA ESA Message Format Service. The edit exit example USRME001 in Figure 42 on page 132 has no functional code as it is only an example of the DSLGEN process.

It consists of two parts:

1. Installation of the user modification with SMP/E
2. The DSLGEN process for the user modification

A sample coded edit exit (DSLME899) is supplied with the machine-readable material and can be used as a coding example.

```
//* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//* Install User Modification: Add MFS Edit Exit Program USRME001
//*                            Add MFS Program Table Copy Code USRMPTTC
//* SYSMOD ID = UUME001
//* Receive and apply the usermod / perform JCLIN
//* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//USERMOD EXEC PGM=GIMSMP,REGION=4M,
//             PARM='CSI=MERVA.SMPCSI.CSI'
//SYSLIB   DD DSN=&PRFX..SDSLMAC0,DISP=SHR,DCB=BLKSIZE=23440
//         DD DSN=SYS1.MACLIB,DISP=SHR
//SMPHOLD  DD  *
++NULL.
//SMPCNTL  DD  *
SET BOUNDARY (GLOBAL).
 RECEIVE.
SET BOUNDARY (DSLTGT) OPTIONS(DSLOPTN).
 JCLIN .
 APPLY S(UUME001).
/*
//SMPPTFIN DD *
++USERMOD(UUME001).
++VER(Z038) FMID(H0B0410).
++MAC(USRMPTTC) DISTLIB(ADSLMAC0) SYSLIB(SDSLMAC0).
* Copy Code for definitions of MFS exits written by user.
* Add the MFS edit exit program USRME001 to MFS.
         DSLMPT NAME=USRME001,NUMBER=001,TYPE=E
*
++SRC(USRME001) DISTLIB(ADSLSRC0) SYSLIB(SDSLSRC0) DISTMOD(ADSLMOD0).
* Dummy MFS edit exit to show the DSLGEN process.
USRME001 DSLMFS MF=START,MODNUM=001,TYPE=EDIT,OPT=(REASON,EXTTS)
         B     MFSGOOD             RETURN TO COMMON EXIT POINT
         LTORG ,
MFSTS    DSECT                     RESUME WORKING STORAGE DSECT
MFSTTSLL EQU   *-MFSTPREF          LENGTH DEFINITION (REQUIRED!)
         END
/*
//SMPJCLIN DD DATA,DLM=@#
//DSLJCLIN JOB (ACCT),NAME,MSGCLASS=L,CLASS=Z
//DSLMMFS  EXEC PGM=HEWL,PARM='RENT,AMODE=31,RMODE=ANY'
//ADSLMOD0 DD DSN=MERVA.ADSLMOD0,DISP=SHR
//SYSLMOD  DD DSN=MERVA.SDSLLODB,DISP=SHR
//SYSLIN   DD  *
  ENTRY DSLMMFS
  ORDER DSLMMFS
  INCLUDE ADSLMOD0(USRME001)
  INCLUDE ADSLMOD0(DSLMMFS)
  NAME DSLMMFS(R)
//
@#
//
```

Figure 42. Install MFS User-Exit Program with SMP/E

SMP/E copies USRMPTTC into the target macro library MERVA.SDSLMAC0, and USRME001 into the target source library MERVA.SDSLSRC0. USRME001 is assembled, and the resulting object deck link-edited to the load module DSLMMFS.

To use the new edit exit with MFS, it is defined in the copy code USRMPTTC, which is added to the MFS program table DSLMPTT in the DSLGEN process. Figure 43 on page 133 shows the DSLGEN process for a user copy code.

```
//*********************************************************************
//* MERVA ESA TABLE GENERATION PROCESS
//*********************************************************************
//STAGE1   PROC PRFX='MERVA'                                    MODIFY
//GENTASM  EXEC PGM=ASMA90,REGION=1024K,
//         PARM='SYSPARM(MVS),DECK,NOOBJECT'
//SYSLIB   DD    DSN=&PRFX..SDSLGEN0,DISP=SHR,DCB=BLKSIZE=23440
//         DD    DSN=&PRFX..SDSLMAC0,DISP=SHR
//         DD    DSN=SYS1.MACLIB,DISP=SHR
//SYSUT1   DD   UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD   SYSOUT=*
//SYSPUNCH DD   DSN=&PRFX..SDSLGEN0(&MBR),DISP=SHR
//SYSIN    DD   DDNAME=SYSIN
//IEBUPDTE EXEC PGM=IEBUPDTE,PARM=NEW,COND=(4,LE)
//SYSPRINT DD    SYSOUT=*
//SYSUT1   DD    DSN=&PRFX..SDSLGEN0,DISP=SHR
//SYSUT2   DD    DSN=&PRFX..SDSLGEN0,DISP=SHR
//SYSIN    DD    DSN=&PRFX..SDSLGEN0(&MBR),DISP=SHR
//         PEND
//*********************************************************************
//* MERVA ESA STAGE 1      GENERATE MERVA ESA TABLES AND SMP/E CONTROL
//*******************************************************************
**
//DSLGENT  EXEC  STAGE1,MBR=DSLGENT
//SYSIN    DD    *
GENT     TITLE 'MERVA ESA and SWIFT Link MVS STAGE 1 GENERATION'
DSLGENT  DSLGEN TYPE=INITIAL,ID=(SWIFT),                        MODIFY *
               EPT=NO,                                                 *
               FDT=NO,                                                 *
               FLT=NO,                                                 *
               FNT=NO,                                                 *
               MCM=NO,                                                 *
               MPT=YES,         ONLY DSLMPTT IS GENERATED              *
               MSG=NO,                                                 *
               MTT=NO,                                                 *
               NCM=NO,                                                 *
               NPT=NO,                                                 *
               NTR=NO
         DSLGEN TYPE=USER,MPT=YES     USER COPY CODE USRMPTTC
         DSLGEN TYPE=FINAL
         END
//*********************************************************************
//* MERVA ESA STAGE 2      INSTALL MERVA ESA TABLES AS SMP/E USERMODS
//*********************************************************************
//MERVASMP EXEC PGM=GIMSMP,REGION=4M,
//             PARM='CSI=MERVA.SMPCSI.CSI',COND=(4,LE),PRFX='MERVA'
//SDSLGEN0 DD    DSN=&PRFX..SDSLGEN0,DISP=SHR
//SMPPTFIN DD    DSN=&PRFX..SDSLGEN0(SMPPTFIN),DISP=SHR
//SMPCNTL  DD    DSN=&PRFX..SDSLGEN0(SMPCNTL),DISP=SHR
//SMPHOLD  DD    DSN=&PRFX..SDSLGEN0(SMPHOLD),DISP=SHR
//
```

Figure 43. MERVA ESA DSLGEN Process for a User Copy Code

The only table generated by this job is DSLMPTT. The table contains the statement:

```
COPY  USRMPTTC
```

This is in addition to the copy statements for the MERVA ESA and the SWIFT Link definitions.

Stage 2 of this job installs the generated DSLMPTT as SMP/E USERMOD UDSLMPT. SMP/E link-edits this table into DSLMMFS.

The same process can be used to add source modules to the MERVA ESA programs DSLEUD and DSLNUC, for example, command execution routines.

The process to add definitions to the MERVA ESA tables DSLFDTT, DSLFLTT, DSLFNTT, DSLMSGT, or DSLMTTT is similar. Only the ++SCR together with the source and JCLIN are omitted in the installation of the user modification with SMP/E, because no modules are link-edited to these tables.

Loading an MFS User Exit Instead of Link-Editing It

The following example shows how to load MFS user exits at execution time instead of link-editing them to DSLMMFS. The MFS edit exits 903 and 904 are changed in the copy code DSLMPTTC. Only the two changed lines are shown in Figure 44.

```
//* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//* Install User Modification: MFS Program Table Copy Code DSLMPTTC
//* SYSMOD ID = UUMPT01
//* Receive and apply the usermod
//* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
//MERVASMP EXEC PGM=GIMSMP,REGION=4M,
//              PARM='CSI=MERVA.SMPCSI.CSI',PRFX='MERVA'
//SYSLIB   DD DSN=&PRFX..SDSLMAC0,DISP=SHR,DCB=BLKSIZE=23440
//         DD DSN=SYS1.MACLIB,DISP=SHR
//SMPHOLD  DD  *
++NULL.
//SMPCNTL  DD  *
SET BOUNDARY (GLOBAL).
 RECEIVE.
SET BOUNDARY (DSLTGT) OPTIONS(DSLOPTN).
 APPLY S(UUMPT01).
/*
//SMPPTFIN DD *
++USERMOD(UUMPT01).
++VER(Z038) FMID(H0B0410).
++MAC(DSLMPTTC) DISTLIB(ADSLMAC0) SYSLIB(SDSLMAC0) ASSEM(DSLMPTT).
       ....
         DSLMPT NAME=DSLME903,NUMBER=903,TYPE=E,LINK=NO
         DSLMPT NAME=DSLME904,NUMBER=904,TYPE=E,LINK=NO
       ....
//
```

Figure 44. User Modification of DSLMPTT

SMP/E copies DSLMPTTC into the target macro library MERVA.SDSLMAC0. The ASSEM parameter instructs SMP/E to assemble the source member DSLMPTT. The resulting object deck is link-edited to load module DSLMMFS. A DSLGEN process is not necessary as no new table copy codes are introduced.

The MFS edit exits 903 and 904 are still link-edited to the load module DSLMMFS, because the modification in the DSLMPTT does not remove the exits from the load module DSLMMFS. However, these versions of the exits, link-edited to DSLMMFS, are never called because the definition in the DSLMPTT tells DSLMMFS to load these exits from the load library.

Note: If MERVA ESA is running under CICS, you must define the names of the new exits in the CICS program definitions.

The same technique can be used to modify other table copy books, for example, to define alias command names for MERVA ESA operator commands in DSLNCMTC.

Note: If an error occurs in the user modification it can be restored using SMP/E RESTORE command. SMP/E installs the table module from the distribution library ADSLMOD0 into the target library. The changes installed with the previous DSLGEN process are also restored and removed from the target library. Either the DSLGEN process for these tables must be repeated, or the user modification must be corrected and repeated.

User exits or tables that are loaded by MERVA ESA can reside in a private load library where they can be installed without using SMP/E. This applies to the following modules and tables:

- DSLPRM
- DSLFDTT
- DSLFLTT
- DSLFNTT
- DSLGRPT
- DSLKPROC
- DSLMSGT
- DSLMTTT
- DSLNSVT
- DSLTFDT
- DSLTXTT
- Routing tables
- Message Control Blocks (MCBs)
- DWSPRM
- DWSLTT
- DWSLOG2
- DSLMUnnn
- ENLPRM
- EKAPT
- EKAMUnnn

Note: MFS user exit DSLMU054 is link-edited to DSLMMFS for performance reasons.

In this case the private load library must be defined in the STEPLIB DD statement (IMS or batch programs) or in the DFHRPL DD statement (CICS) before the MERVA ESA load library.

If user exits or tables are assembled without SMP/E, the SMP/E target macro library MERVA.SDSLMAC0 must be defined in the SYSLIB DD statement to include all applied service fixes during the assembly.

Table Generation JCL for MERVA ESA (Stage I)

Figure 45 on page 136 shows the JCL for DSLGEN Stage I in MVS.

```
//*********************************************************************
//* MERVA ESA TABLE GENERATION PROCESS
//*********************************************************************
//STAGE1   PROC PRFX='MERVA'                                    MODIFY
//GENTASM  EXEC PGM=ASMA90,REGION=1024K,
//         PARM='SYSPARM(MVS),DECK,NOOBJECT'
//SYSLIB   DD    DSN=&PRFX..SDSLGEN0,DISP=SHR,DCB=BLKSIZE=23440
//         DD    DSN=&PRFX..SDSLMAC0,DISP=SHR
//         DD    DSN=SYS1.MACLIB,DISP=SHR
//SYSUT1   DD   UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD   SYSOUT=*
//SYSPUNCH DD   DSN=&PRFX..SDSLGEN0(&MBR),DISP=SHR
//SYSIN    DD   DDNAME=SYSIN
//IEBUPDTE EXEC PGM=IEBUPDTE,PARM=NEW,COND=(4,LE)
//SYSPRINT DD    SYSOUT=*
//SYSUT1   DD    DSN=&PRFX..SDSLGEN0,DISP=SHR
//SYSUT2   DD    DSN=&PRFX..SDSLGEN0,DISP=SHR
//SYSIN    DD    DSN=&PRFX..SDSLGEN0(&MBR),DISP=SHR
//         PEND
//*
//*********************************************************************
//* MERVA ESA STAGE 1      GENERATE MERVA ESA TABLES AND SMP/E CONTROL
//*********************************************************************
//DSLGENT  EXEC  STAGE1,MBR=DSLGENT
//SYSIN    DD    *
GENT     TITLE 'MERVA ESA STAGE 1 GENERATION'
DSLGENT  DSLGEN TYPE=INITIAL,ID=SWIFT                           MODIFY
         DSLGEN TYPE=FINAL          END
//*
```

Figure 45. JCL for DSLGEN Stage I in MVS

Note: If a USERMOD is used to modify a macro or copy code, macro replacement (++MAC SMP/E control statement) must be used in the USERMOD.

Table Generation JCL for MERVA ESA (Stage II)

Figure 46 shows the JCL for DSLGEN Stage II in MVS.

```
//*********************************************************************
//* MERVA ESA STAGE 2      INSTALL MERVA ESA TABLES AS SMP/E USERMODS
//*********************************************************************
//MERVASMP EXEC PGM=GIMSMP,REGION=4M,
//            PARM='CSI=MERVA.SMPCSI.CSI',COND=(4,LE),PRFX='MERVA'
//SDSLGEN0 DD    DSN=&PRFX..SDSLGEN0,DISP=SHR
//SMPPTFIN DD    DSN=&PRFX..SDSLGEN0(SMPPTFIN),DISP=SHR
//SMPCNTL  DD    DSN=&PRFX..SDSLGEN0(SMPCNTL),DISP=SHR
//SMPHOLD  DD    DSN=&PRFX..SDSLGEN0(SMPHOLD),DISP=SHR
//
```

Figure 46. JCL for DSLGEN Stage II in MVS

Appendix C. Protocol of the Communication with Headoffice Telex on a Fault-Tolerant System

This appendix describes the communication protocol between the Telex Link and Headoffice Telex on a fault-tolerant system. Figure 47 on page 144 shows an example session between the Telex Link and Headoffice Telex on a fault-tolerant system.

The protocol assumes an asynchronous two-way communication, where traffic in either direction is synchronized by acknowledgments for every important piece of information transmitted. Periods in the following protocol formats indicate blanks in the exchanged data.

Traffic from the Telex Link to the Telex Station

Signon Command

The Telex Link starts a session with the Headoffice Telex on a fault-tolerant system with the signon command:

@_mmmmmmmm_**........SS001**_bbbbbllll_**.**_yymmddhhmmss_

This command is composed as follows: [4]

@ and **SS001**
Indicates the signon command.

mmmmmmmm
Is the station name as defined in the customizing parameters ENLPRM. This name is used in each header of communicated data.

bbbbb
Is the buffer size limiting the size of telex messages in the communication between the Telex Link and Headoffice Telex on a fault-tolerant system (BUFSIZE parameter of the customizing parameters ENLPRM).

llll
Is the maximum number of text lines that can be transmitted from Headoffice Telex on a fault-tolerant system to the Telex Link. A value of 9999 lines is used.

yymmddhhmmss
Is the actual date and time for the Telex Link. Headoffice Telex on a fault-tolerant system will use this date and time to synchronize the communication between the two.

Signon Acknowledgment

The Headoffice Telex on a fault-tolerant system acknowledges the signon command coming from the Telex Link with a positive or negative signon acknowledgment.

The positive signon acknowledgment has the following format:

[4] Symbols used in this protocol are only described once.

@*mmmmmmmmmssss***....SS001..........ACK**,**SS00**,*xxxxx***.**

Where the components are:

ACK,SS00
Indicates the signon acknowledgment.

ssss
Is the session number for the current session (which is one greater than the previous one; wrap-around occurs at 9999).

xxxxx
Is the status of Headoffice Telex on a fault-tolerant system at the time it received the signon message.

The negative signon acknowledgment has the following formats:

@*mmmmmmmmmssss***....SS001..........NAK**,**SS21**,*xxxxx***.**

NAK,SS21
Indicates that a session is signed on already.

@*mmmmmmmmmssss***....SS001..........NAK**,**SS01**,*xxxxx***.**

NAK,SS01
Indicates that the buffer size contained in the signon command is too big for Headoffice Telex on a fault-tolerant system. You must use a smaller value in the BUFSIZE parameter in the customizing parameters ENLPRM.

Signoff Command

The Telex Link ends a session with the Headoffice Telex on a fault-tolerant system with the signoff command:

@*mmmmmmmmmssss***....SS002..........**

@ and **SS002**
Indicates the signoff command.

Signoff Acknowledgment

The Headoffice Telex on a fault-tolerant system always acknowledges the signoff command coming from the Telex Link with a positive signoff acknowledgment, regardless of whether or not there is a session.

@*mmmmmmmmmssss***....SS002..........ACK**,**SS00**,*xxxxx***.**

ACK,SS00
Indicates the signoff acknowledgment.

Outgoing Telex Message

A telex message sent from the Telex Link to the Headoffice Telex on a fault-tolerant system contains the following header:

'*mmmmmmmmmssssnnnn***MS001***sssOnnnO***..**

This header is composed as follows:

' and **MS001**
Indicates a message sent from the Telex Link to the Headoffice Telex on a fault-tolerant system.

ssssnnnn
Are the current session and sequence numbers. In a session, the sequence number starts with 0000 for the first telex message sent to the Headoffice Telex

on a fault-tolerant system, and is incremented by 1 for each subsequent telex message.

sssOnnnO
Are the session and sequence numbers of the first attempt to send this telex message from the Telex Link to Headoffice Telex on a fault-tolerant system. *sssOnnnO* is also called the “original reference”. *sssOnnnO* can only be different from *ssssnnnn* for the first message of a session (that is, *nnnn* is 0000) if the Telex Link did not receive the logical acknowledgment for the last telex message sent to Headoffice Telex on a fault-tolerant system in the previous session.

Logical acknowledgments and transmission acknowledgments refer to the original reference.

Logical Acknowledgment

Each telex message sent from the Telex Link to the Headoffice Telex on a fault-tolerant system is acknowledged by a positive or negative logical acknowledgment.

The format of the positive logical acknowledgment is:

'*mmmmmmmmmssssnnnn***MS001***sssOnnnO***..ACK**,**NS00**,*d*,*info-1*

The logical acknowledgment consists of a copy of the telex message header followed by:

ACK,NS00
Indicates a positive logical acknowledgment.

d
Indicates whether Headoffice Telex on a fault-tolerant system received this telex message a second time (that is, it is a duplicate, and *d*=2), or whether Headoffice Telex on a fault-tolerant system received this telex message for the first time (*d*=0).

info-1
Indicates the transmission data information as applicable to logical acknowledgments (see “Telex Transmission Data” on page 144).

The formats of the negative logical acknowledgment are:

'*mmmmmmmmmssssnnnn***MS001***sssOnnnO***..NAK**,**AA01.**

NAK,AA01
Indicates the negative logical acknowledgment for incorrect session or sequence numbers.

'*mmmmmmmmmssssnnnn***MS001***sssOnnnO***..NAK**,**AA02**

NAK,AA02
Indicates the negative logical acknowledgment for incorrect format of the telex header or incorrect data in the telex text.

In some cases, a negative logical acknowledgment is internally generated by the Telex Link:

'*mmmmmmmmmssssnnnn***MS001***sssOnnnO***..NAK**,**FMT**,*reason-text*

NAK,FMT Indicates that the Telex Link did not send the telex message to Headoffice Telex on a fault-tolerant system because it found errors when preparing the telex message for transmission.

reason-text Indicates the reason for not sending the telex message with an error message starting with ENL followed by a number (refer to *MERVA for ESA Messages and Codes*).

Possible reasons are:

- The MERVA ESA Message Format Service found an error when formatting the telex message to the telex line format.
- Incorrect tags were found for the telex transmission information.
- The message is larger than the buffer size agreed upon with the Headoffice Telex on a fault-tolerant system.

Traffic from the Telex Station to the Telex Link

Transmission Acknowledgments

When a message has been successfully sent to the receiver via the telex network, a positive or negative transmission acknowledgment is sent from the Headoffice Telex on a fault-tolerant system to the Telex Link.

The format of the positive transmission acknowledgment is:

!*mmmmmmmmssss***....RR002***sssOnnnO***..ACK**,*info-2*

Where the components are:

! and **RR002**
Identify the transmission acknowledgment.

mmmmmmmm
Is the station name as agreed upon with Headoffice Telex on a fault-tolerant system during signon.

ssss Is the current session number.

sssOnnnO
Are the original session and sequence numbers of the telex message (original reference).

ACK Indicates the positive transmission acknowledgment.

info-2 Indicates the transmission data information as applicable to positive transmission acknowledgments (see “Telex Transmission Data” on page 144). This includes, for example, the transmission time and the duration of transmission.

The format of the negative transmission acknowledgment is:

!*mmmmmmmmssss***....RR002***sssOnnnO***..NAK**,*info-3*

NAK Indicates the negative transmission acknowledgment.

info-3
Indicates the reason for the NAK with the transmission data information as applicable to negative transmission acknowledgments (see "Telex Transmission Data" on page 144).

Both positive and negative transmission acknowledgments are in turn acknowledged by the Telex Link:

!*mmmmmmmmmssss***....RR002***sssOnnnO***..ACK.**

ACK
Indicates that the Telex Link acknowledges the transmission acknowledgment.

Received Telex Messages

Each telex message sent from the Headoffice Telex on a fault-tolerant system to the Telex Link has the following header:

$*mmmmmmmmmssssnnnn***SM001***sssOnnnOblinfo-4***.***message*

$ and **SM001**
Indicates a received telex message that is sent from Headoffice Telex on a fault-tolerant system to the Telex Link.

ssssnnnn
Are the current session and sequence numbers. In a session, the sequence number starts with 0000 for the first telex message sent by Headoffice Telex on a fault-tolerant system to the Telex Link, and is incremented by 1 for each subsequent telex message sent by Headoffice Telex on a fault-tolerant system to the Telex Link.

sssOnnnO
Are the session and sequence number of the first attempt to send this telex message from Headoffice Telex on a fault-tolerant system to the Telex Link. *sssOnnnO* is also called the "original reference". *sssOnnnO* can only be different from *ssssnnnn* for the first message of a session (that is, *nnnn* is 0000), if Headoffice Telex on a fault-tolerant system did not receive the logical acknowledgment for the last telex message sent to the Telex Link in the previous session.

b
b=**B** indicates that the telex message was truncated before it was sent to the Telex Link, because it is larger than the buffer agreed upon during signon. If the telex message is not truncated, this indicator is blank.

l
l=**L** indicates that the telex message was truncated before it was sent to the Telex Link, because it contains more than the 9999 lines agreed upon during signon. If the telex message is not truncated, this indicator is blank.

info-4
Indicates the transmission data information as applicable to received telex messages (see "Telex Transmission Data" on page 144).

message
Is the received telex message. Lines are separated

by carriage-return and line-feed characters (X'0D25'). A padding character is inserted between *info-4* and the message.

Each telex message sent from the Headoffice Telex on a fault-tolerant system to the Telex Link is confirmed by an acknowledgment of the following format:

$*mmmmmmmmssssnnnn***SM001***sssOnnnObl***ACK.**

The acknowledgment consists of a copy of the telex message header followed by:

ACK
Indicates that the Telex Link acknowledges the reception of the received telex message.

If the Telex Link finds more than the 9999 lines agreed upon during signon in the received telex message, it accepts the telex message but sets an indicator “S” into the acknowledgment:

$*mmmmmmmmssssnnnn***SM001***sssOnnnO***.SACK.**

SACK
Indicates that the number of lines was exceeded.

Status Report

The Headoffice Telex on a fault-tolerant system periodically sends a status report to the Telex Link (every two minutes):

!*mmmmmmmmssss***....RR999..........***status*

! and **RR999**
Indicate the status report.

status
Offset 14...17 of the status report reflects the status code as shown in “Telex Interface Program Status Codes” on page 145.

These status reports are *not* acknowledged by the Telex Link.

General Negative Acknowledgment

When receiving invalid data, both the Telex Link and the Headoffice Telex on a fault-tolerant system send a general negative acknowledgment:

*cmmmmmmmmssss***....NAK99..........***received-data*

NAK99
Indicates the general negative acknowledgment.

c
Indicates whether the negative acknowledgment was sent by Headoffice Telex on a fault-tolerant system or by the Telex Link. *c* is **!** when the negative acknowledgment is sent from Headoffice Telex on a fault-tolerant system to the Telex Link; *c* is **@** when the negative acknowledgment is sent from the Telex Link to Headoffice Telex on a fault-tolerant system.

received-data
Contains up to 32 bytes of the invalid received data.

No-Operation Data

In some error situations, the Telex Link sends a “no-operation” message to the Headoffice Telex on a fault-tolerant system:

@*mmmmmmmmssss***....NOP99..........**

@ and **NOP99** Indicate the “no-operation” message.

Headoffice Telex on a fault-tolerant system ignores this message.

Example of a Telex Link via a Fault-Tolerant System Session

Figure 47 on page 144 shows an example of a session between the Telex Link and Headoffice Telex on a fault-tolerant system.

```
Operation          Telex Link                     |                Telex Station
_________________________________________________________________________________
                                                  |
Signon             Signon command           ----SS001--->
                                            <---SS001----       Signon ACK | NAK
                                                  |
                   ______________________________________________________________
                                                  |
Outgoing           Outgoing telex message   ----MS001--->
telex                                       <---MS001----      Logical ACK | NAK
message                                           |            (after received by
                                                  |            the Telex Station)
                                                  |
                                            <---RR002----  Transmission ACK | NAK
                                                  |        (after transmission to
                                                  |        the telex network was
                                                  |        successful or failed)
                                                  |
                   Acknowledgment to        ----RR002--->
                   transmission ACK | NAK         |
                                                  |
                   ______________________________________________________________
                                                  |
Incoming                                    <---SM001----  Incoming telex message
telex              Acknowledgment           ----SM001--->
                   ACK | SACK                     |
                                                  |
                   ______________________________________________________________
                                                  |
Status report                               <---RR999----           Status report
                   (No acknowledgment)            |
                                                  |
                   ______________________________________________________________
                                                  |
Signoff            Signoff command          ----SS002--->
                                            <---SS002----            Signoff ACK
                                                  |
                   ______________________________________________________________
                                                  |
General negative                            ----NAK99--->              (In either
acknowledgment                              <---NAK99----              direction)
                                                  |
                   ______________________________________________________________
                                                  |
No-Operation                                ----NOP99--->
(NOP) message                                     |
                                                  |
```

Figure 47. Example of a Telex Link - Headoffice Telex on a fault-tolerant system Session

Telex Transmission Data

The Headoffice Telex on a fault-tolerant system provides transmission data when sending the following messages to the Telex Link:

info-1 Positive logical acknowledgment

info-2 Positive transmission acknowledgment

info-3 Negative transmission acknowledgment

info-4 Received telex messages

The term *info-x* is used in the description of these messages in the previous section of this appendix. This information has always the same layout, but the fields are filled only as shown in Figure 48 for the four types of info-x. The contents of unused fields are unpredictable.

	info-					
1	2	3	4	Length	Field	Meaning
	X	X		20	TXNUMR	Network response
X	X	X	X	5	TXISEQ	Document number
	X		X	5	TXCOUNT	Telex count: sent or received
	X	X		12	TXATIME	Time & date of last access to this index rec
	X			12	TXETIME	Time & date of transmission end
	X	X		2	TXSTATUS	Telex status (stored, tried, sent)
	X			2	TXRETRY	Number of retries
				10	Reserved	
	X	X		2	TXSLINE	Selected sending line number
	X			4	TXDURM	Duration in minutes
	X			2	TXDURS	Rest of duration in seconds
				18	Reserved	
			X	2	TRLINE	Line number on which received
	X	X	X	12	TRTIME	Telex time stamp (HHMMSSMMDDYY)
				1	Reserved	
	X			5	TXSPSEQ	Sequence numberfor split telex
				6	Reserved	

Figure 48. Fields of the Telex Transmission Data

An **X** in the info number column for a field indicates that this field is used in that type of information.

Telex Interface Program Status Codes

The Headoffice Telex on a fault-tolerant system reports its status in two-minute intervals to the Telex Link. The status codes used are shown in the following list.

Code **Meaning**

0050 Waiting for signon after IPL on the fault-tolerant front end

0090 Waiting for signon after internal signoff resulting from an error situation, for example, a line or SNA error

0100 Disconnected idle (no session active)

0110 Sending VTAM logon

0200 Signon received, about to send signon ACK

0210 Connected idle

0220 Disconnect pending, about to send signoff ACK

0320 Message received from the Telex Link, checking in progress

0330 Message received from the Telex Link, formatting in progress

0340 Message received from the Telex Link, about to send acknowledgment

0350 Telex received from host, formatting in progress, and a telex found for host; a quickly-replaced, temporary intermediate status

0410 Connected idle, about to send message to the Telex Link

0411 Connected idle, about to send transmission acknowledgment to the Telex Link

0420 Connected, waiting for acknowledgment of message

0421 Connected, waiting for acknowledgment of transmission acknowledgment

0900 Severe error, waiting for signoff

Appendix D. List of Reentrant and Reusable Modules

Some of the MERVA ESA modules can be used in the MVS LPA or the VSE SVA because they are reentrant. Under MVS, all modules in the distribution libraries have the attribute “reentrant” to enable the attribute “reentrant” for the modules in the load library.

Reentrant Modules of the Base Functions

The following modules are reentrant in MERVA ESA:

DSLAF...	MERVA Message Processing Client/Server
DSLAPCIC	Application Interface (CICS only)
DSLCXT	Checking and Expansion Program (CICS only)
DSLEUD	End-User Driver Program (CICS only)
DSLFDTT	Field Definition Table
DSLGRPT	Group Table
DSLHCP	Hardcopy Printer Program (CICS only)
DSLK.....	MERVA-MQI Attachment Programs and Tables
DSLMMFS	Message Format Service (see note)
DSLMSGT	Message Table
DSLMTTT	Message Type Table
DSLNSVT	Nucleus Server Table
DSLPRM	Customizing Parameter Module
DSLPTMRV	Program-to-MERVA Switch (CICS only)
DSLQMGD	Queue management for DB2
DSLTFDT	Terminal Feature Definition Table
DSLTOFSV	TOF Supervisor Program (see note)
DSL.....	All Message Control Blocks (MCBs)

Note: When the debugging trace of MERVA ESA or the debugging command TSHOT is used, the programs DSLMMFS and DSLTOFSV are not reentrant and must not be loaded into the MVS LPA or the VSE SVA. The debugging trace is described in the *MERVA for ESA Diagnosis Guide*.

Reentrant Modules of the SWIFT Link

The following modules are reentrant in the SWIFT Link:

DWSnnn	All Message Control Blocks (MCBs)
DWSRxxxx	All routing modules
DWSLxxxx	All routing modules
DWSMCFLH	Checking and Separation module for SWIFT fields

DWSMCMLH	Checking and Separation module for SWIFT messages
DWSNAEVV	SWIFT line subtask module

Reentrant Modules of the Telex Link

The following modules are reentrant in the Telex Link via a fault-tolerant system:

ENLHCF1	Telex Substation Interface Program (CICS only)
ENLPRM	Customizing Parameter Module

In the Telex Link, all Message Control Blocks (MCBs) and routing modules are reentrant.

Reentrant Modules of the MERVA Link

The following modules are reentrant in MERVA Link:

EKASP10	Message Transfer Service Processor
EKATM10	Back-to-Back Mirror
EKATPO1	Sending MTP (APPC/MVS)
EKATPI1	Receiving MTP (APPC/MVS)
EKATS10	Sending MTP (CICS)
EKATR10	Receiving MTP (CICS)

Appendix E. 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

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, other countries, or both:

- Advanced Peer-to-Peer Networking
- AIX
- APPN
- C/370
- CICS
- CICS/ESA
- CICS/MVS
- CICS/VSE
- DB2
- DB2 Universal Database
- Distributed Relational Database Architecture
- DRDA
- IBM
- IMS/ESA
- Language Environment
- MQSeries

- MVS
- MVS/ESA
- MVS/XA
- OS/2
- OS/390
- RACF
- VisualAge
- 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.

Glossary of Terms and Abbreviations

This glossary defines terms as they are used in this book. If you do not find the terms you are looking for, refer to the *IBM Dictionary of Computing*, New York: McGraw-Hill, and the *S.W.I.F.T. User Handbook*.

A

ACB. Access method control block.

ACC. MERVA Link USS application control command application. It provides a means of operating MERVA Link USS in USS shell and MVS batch environments.

Access method control block (ACB). A control block that links an application program to VSAM or VTAM.

ACD. MERVA Link USS application control daemon.

ACT. MERVA Link USS application control table.

address. See *SWIFT address*.

address expansion. The process by which the full name of a financial institution is obtained using the SWIFT address, telex correspondent's address, or a nickname.

AMPDU. Application message protocol data unit, which is defined in the MERVA Link P1 protocol, and consists of an envelope and its content.

answerback. In telex, the response from the dialed correspondent to the WHO R U signal.

answerback code. A group of up to 6 letters following or contained in the answerback. It is used to check the answerback.

APC. Application control.

API. Application programming interface.

APPC. Advanced Program-to-Program Communication based on SNA LU 6.2 protocols.

APPL. A VTAM definition statement used to define a VTAM application program.

application programming interface (API). An interface that programs can use to exchange data.

application support filter (ASF). In MERVA Link, a user-written program that can control and modify any data exchanged between the Application Support Layer and the Message Transfer Layer.

application support process (ASP). An executing instance of an application support program. Each application support process is associated with an ASP entry in the partner table. An ASP that handles outgoing messages is a *sending ASP*; one that handles incoming messages is a *receiving ASP*.

application support program (ASP). In MERVA Link, a program that exchanges messages and reports with a specific remote partener ASP. These two programs must agree on which conversation protocol they are to use.

ASCII. American Standard Code for Information Interchange. The standard code, using a coded set consisting of 7-bit coded characters (8 bits including parity check), used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

ASF. Application support filter.

ASF. (1) Application support process. (2) Application support program.

ASPDU. Application support protocol data unit, which is defined in the MERVA Link P2 protocol.

authentication. The SWIFT security check used to ensure that a message has not changed during transmission, and that it was sent by an authorized sender.

authenticator key. A set of alphanumeric characters used for the authentication of a message sent via the SWIFT network.

authenticator-key file. The file that stores the keys used during the authentication of a message. The file contains a record for each of your financial institution's correspondents.

B

Back-to-Back (BTB). A MERVA Link function that enables ASPs to exchange messages in the local MERVA Link node without using data communication services.

bank identifier code. A 12-character code used to identify a bank within the SWIFT network. Also called a SWIFT address. The code consists of the following subcodes:
- The bank code (4 characters)
- The ISO country code (2 characters)
- The location code (2 characters)
- The address extension (1 character)

- The branch code (3 characters) for a SWIFT user institution, or the letters “BIC” for institutions that are not SWIFT users.

Basic Security Manager (BSM). A component of VSE/ESA Version 2.4 that is invoked by the System Authorization Facility, and used to ensure signon and transaction security.

BIC. Bank identifier code.

BIC Bankfile. A tape of bank identifier codes supplied by S.W.I.F.T.

BIC Database Plus Tape. A tape of financial institutions and currency codes, supplied by S.W.I.F.T. The information is compiled from various sources and includes national, international, and cross-border identifiers.

BIC Directory Update Tape. A tape of bank identifier codes and currency codes, supplied by S.W.I.F.T., with extended information as published in the printed BIC Directory.

body. The second part of an IM-ASPDU. It contains the actual application data or the message text that the IM-AMPDU transfers.

BSC. Binary synchronous control.

BSM. Basic Security Manager.

BTB. Back-to-back.

buffer. A storage area used by MERVA programs to store a message in its internal format. A buffer has an 8-byte prefix that indicates its length.

C

CBT. SWIFT computer-based terminal.

CCSID. Coded character set identifier.

CDS. Control data set.

central service. In MERVA, a service that uses resources that either require serialization of access, or are only available in the MERVA nucleus.

CF message. Confirmed message. When a sending MERVA Link system is informed of the successful delivery of a message to the receiving application, it routes the delivered application messages as CF messages, that is, messages of class CF, to an ACK wait queue or to a complete message queue.

COA. Confirm on arrival.

COD. Confirm on delivery.

coded character set identifier (CCSID). The name of a coded set of characters and their code point assignments.

commit. In MQSeries, to commit operations is to make the changes on MQSeries queues permanent. After putting one or more messages to a queue, a commit makes them visible to other programs. After getting one or more messages from a queue, a commit permanently deletes them from the queue.

confirm-on-arrival (COA) report. An MQSeries report message type created when a message is placed on that queue. It is created by the queue manager that owns the destination queue.

confirm-on-delivery (COD) report. An MQSeries report message type created when an application retrieves a message from the queue in a way that causes the message to be deleted from the queue. It is created by the queue manager.

control fields. In MERVA Link, fields that are part of a MERVA message on the queue data set and of the message in the TOF. Control fields are written to the TOF at nesting identifier 0. Messages in SWIFT format do not contain control fields.

correspondent. An institution to which your institution sends and from which it receives messages.

correspondent identifier. The 11-character identifier of the receiver of a telex message. Used as a key to retrieve information from the Telex correspondents file.

cross-system coupling facility. See *XCF*.

coupling services. In a sysplex, the functions of XCF that transfer data and status information among the members of a group that reside in one or more of the MVS systems in the sysplex.

couple data set. See *XCF couple data set*.

CTP. MERVA Link command transfer processor.

currency code file. A file containing the currency codes, together with the name, fraction length, country code, and country names.

D

daemon. A long-lived process that runs unattended to perform continuous or periodic systemwide functions.

DASD. Direct access storage device.

data area. An area of a predefined length and format on a panel in which data can be entered or displayed. A field can consist of one or more data areas.

data element. A unit of data that, in a certain context, is considered indivisible. In MERVA Link, a data element consists of a 2-byte data element length field, a 2-byte data-element identifier field, and a field of variable length containing the data element data.

datagram. In TCP/IP, the basic unit of information passed across the Internet environment. This type of message does not require a reply, and is the simplest type of message that MQSeries supports.

data terminal equipment. That part of a data station that serves as a data source, data link, or both, and provides for the data communication control function according to protocols.

DB2. A family of IBM licensed programs for relational database management.

dead-letter queue. A queue to which a queue manager or application sends messages that it cannot deliver. Also called *undelivered-message queue*.

dial-up number. A series of digits required to establish a connection with a remote correspondent via the public telex network.

direct service. In MERVA, a service that uses resources that are always available and that can be used by several requesters at the same time.

display mode. The mode (PROMPT or NOPROMPT) in which SWIFT messages are displayed. See *PROMPT mode* and *NOPROMPT mode.*

distributed queue management (DQM). In MQSeries message queuing, the setup and control of message channels to queue managers on other systems.

DQM. Distributed queue management.

DTE. Data terminal equipment.

E

EBCDIC. Extended Binary Coded Decimal Interchange Code. A coded character set consisting of 8-bit coded characters.

ECB. Event control block.

EDIFACT. Electronic Data Interchange for Administration, Commerce and Transport (a United Nations standard).

ESM. External security manager.

EUD. End-user driver.

exception report. An MQSeries report message type that is created by a message channel agent when a message is sent to another queue manager, but that message cannot be delivered to the specified destination queue.

external line format (ELF) messages. Messages that are not fully tokenized, but are stored in a single field in the TOF. Storing messages in ELF improves performance, because no mapping is needed, and checking is not performed.

external security manager (ESM). A security product that is invoked by the System Authorization Facility. RACF is an example of an ESM.

F

FDT. Field definition table.

field. In MERVA, a portion of a message used to enter or display a particular type of data in a predefined format. A field is located by its position in a message and by its tag. A field is made up of one or more data areas. See also *data area*.

field definition table (FDT). The field definition table describes the characteristics of a field; for example, its length and number of its data areas, and whether it is mandatory. If the characteristics of a field change depending on its use in a particular message, the definition of the field in the FDT can be overridden by the MCB specifications.

field group. One or several fields that are defined as being a group. Because a field can occur more than once in a message, field groups are used to distinguish them. A name can be assigned to the field group during message definition.

field group number. In the TOF, a number is assigned to each field group in a message in ascending order from 1 to 255. A particular field group can be accessed using its field group number.

field tag. A character string used by MERVA to identify a field in a network buffer. For example, for SWIFT field 30, the field tag is **:30:**.

FIN. Financial application.

FIN-Copy. The MERVA component used for SWIFT FIN-Copy support.

finite state machine. The theoretical base describing the rules of a service request's state and the conditions to state transitions.

FMT/ESA. MERVA-to-MERVA Financial Message Transfer/ESA.

form. A partially-filled message containing data that can be copied for a new message of the same message type.

G

GPA. General purpose application.

H

HFS. Hierarchical file system.

hierarchical file system (HFS). A system for organizing files in a hierarchy, as in a UNIX system. OS/390 UNIX System Services files are organized in an HFS. All files are members of a directory, and each directory is in turn a member of a directory at a higher level in the HFS. The highest level in the hierarchy is the root directory.

I

IAM. Interapplication messaging (a MERVA Link message exchange protocol).

IM-ASPDU. Interapplication messaging application support protocol data unit. It contains an application message and consists of a heading and a body.

incore request queue. Another name for the request queue to emphasize that the request queue is held in memory instead of on a DASD.

InetD. Internet Daemon. It provides TCP/IP communication services in the OS/390 USS environment.

initiation queue. In MQSeries, a local queue on which the queue manager puts trigger messages.

input message. A message that is input into the SWIFT network. An input message has an input header.

INTERCOPE TelexBox. This telex box supports various national conventions for telex procedures and protocols.

interservice communication. In MERVA ESA, a facility that enables communication among services if MERVA ESA is running in a multisystem environment.

intertask communication. A facility that enables application programs to communicate with the MERVA nucleus and so request a central service.

IP. Internet Protocol.

IP message. In-process message. A message that is in the process of being transferred to another application.

ISC. Intersystem communication.

ISN. Input sequence number.

ISN acknowledgment. A collective term for the various kinds of acknowledgments sent by the SWIFT network.

ISO. International Organization for Standardization.

ITC. Intertask communication.

J

JCL. Job control language.

journal. A chronological list of records detailing MERVA actions.

journal key. A key used to identify a record in the journal.

journal service. A MERVA central service that maintains the journal.

K

KB. Kilobyte (1024 bytes).

key. A character or set of characters used to identify an item or group of items. For example, the user ID is the key to identify a user file record.

key-sequenced data set (KSDS). A VSAM data set whose records are loaded in key sequence and controlled by an index.

keyword parameter. A parameter that consists of a keyword, followed by one or more values.

KSDS. Key-sequenced data set.

L

LAK. Login acknowledgment message. This message informs you that you have successfully logged in to the SWIFT network.

large message. A message that is stored in the large message cluster (LMC). The maximum length of a message to be stored in the VSAM QDS is 31900 bytes. Messages up to 2MB can be stored in the LMC. For queue management using DB2 no distinction is made between messages and large messages.

large queue element. A queue element that is larger than the smaller of:

- The limiting value specified during the customization of MERVA
- 32KB

LC message. Last confirmed control message. It contains the message-sequence number of the application or acknowledgment message that was last confirmed; that is, for which the sending MERVA Link system most recently received confirmation of a successful delivery.

LDS. Logical data stream.

LMC. Large message cluster.

LNK. Login negative acknowledgment message. This message indicates that the login to the SWIFT network has failed.

local queue. In MQSeries, a queue that belongs to a local queue manager. A local queue can contain a list of messages waiting to be processed. Contrast with *remote queue*.

local queue manager. In MQSeries, the queue manager to which the program is connected, and that provides message queuing services to that program. Queue managers to which a program is not connected are remote queue managers, even if they are running on the same system as the program.

login. To start the connection to the SWIFT network.

LR message. Last received control message, which contains the message-sequence number of the application or acknowledgment message that was last received from the partner application.

LSN. Login sequence number.

LT. See *LTERM*.

LTC. Logical terminal control.

LTERM. Logical terminal. Logical terminal names have 4 characters in CICS and up to 8 characters in IMS.

LU. A VTAM logical unit.

M

maintain system history program (MSHP). A program used for automating and controlling various installation, tailoring, and service activities for a VSE system.

MCA. Message channel agent.

MCB. Message control block.

MERVA ESA. The IBM licensed program Message Entry and Routing with Interfaces to Various Applications for ESA.

MERVA Link. A MERVA component that can be used to interconnect several MERVA systems.

message. A string of fields in a predefined form used to provide or request information. See also *SWIFT financial message.*

message body. The part of the message that contains the message text.

message category. A group of messages that are logically related within an application.

message channel. In MQSeries distributed message queuing, a mechanism for moving messages from one queue manager to another. A message channel comprises two message channel agents (a sender and a receiver) and a communication link.

message channel agent (MCA). In MQSeries, a program that transmits prepared messages from a transmission queue to a communication link, or from a communication link to a destination queue.

message control block (MCB). The definition of a message, screen panel, net format, or printer layout made during customization of MERVA.

Message Format Service (MFS). A MERVA direct service that formats a message according to the medium to be used, and checks it for formal correctness.

message header. The leading part of a message that contains the sender and receiver of the message, the message priority, and the type of message.

Message Integrity Protocol (MIP). In MERVA Link, the protocol that controls the exchange of messages between partner ASPs. This protocol ensures that any loss of a message is detected and reported, and that no message is duplicated despite system failures at any point during the transfer process.

message-processing function. The various parts of MERVA used to handle a step in the message-processing route, together with any necessary equipment.

message queue. See *queue*.

Message Queue Interface (MQI). The programming interface provided by the MQSeries queue managers. It provides a set of calls that let application programs access message queuing services such as sending messages, receiving messages, and manipulating MQSeries objects.

Message Queue Manager (MQM). An IBM licensed program that provides message queuing services. It is part of the MQSeries set of products.

message reference number (MRN). A unique 16-digit number assigned to each message for identification purposes. The message reference number consists of an 8-digit domain identifier that is followed by an 8-digit sequence number.

message sequence number (MSN). A sequence number for messages transferred by MERVA Link.

message type (MT). A number, up to 7 digits long, that identifies a message. SWIFT messages are identified by a 3-digit number; for example SWIFT message type MT S100.

MFS. Message Format Service.

MIP. Message Integrity Protocol.

MPDU. Message protocol data unit, which is defined in P1.

MPP. In IMS, message-processing program.

MQA. MQ Attachment.

MQ Attachment (MQA). A MERVA feature that provides message transfer between MERVA and a user-written MQI application.

MQH. MQSeries queue handler.

MQI. Message queue interface.

MQM. Message queue manager.

MQS. MQSeries nucleus server.

MQSeries. A family of IBM licensed programs that provides message queuing services.

MQSeries nucleus server (MQS). A MERVA component that listens for messages on an MQI queue, receives them, extracts a service request, and passes it via the request queue handler to another MERVA ESA instance for processing.

MQSeries queue handler (MQH). A MERVA component that performs service calls to the Message Queue Manager via the provided Message Queue Interface.

MRN. Message reference number.

MSC. MERVA system control facility.

MSHP. Maintain system history program.

MSN. Message sequence number.

MT. Message type.

MTP. (1) Message transfer program. (2) Message transfer process.

MTS. Message Transfer System.

MTSP. Message Transfer Service Processor.

MTT. Message type table.

multisystem application. (1) An application program that has various functions distributed across MVS systems in a multisystem environment. (2) In XCF, an authorized application that uses XCF coupling services. (3) In MERVA ESA, multiple instances of MERVA ESA that are distributed among different MVS systems in a multisystem environment.

multisystem environment. An environment in which two or more MVS systems reside on one or more processors, and programs on one system can communicate with programs on the other systems. With XCF, the environment in which XCF services are available in a defined sysplex.

multisystem sysplex. A sysplex in which one or more MVS systems can be initialized as part of the sysplex. In a multisystem sysplex, XCF provides coupling services on all systems in the sysplex and requires an XCF couple data set that is shared by all systems. See also *single-system sysplex*.

MVS/ESA. Multiple Virtual Storage/Enterprise Systems Architecture.

N

namelist. An MQSeries for MVS/ESA object that contains a list of queue names.

nested message. A message that is composed of one or more message types.

nested message type. A message type that is contained in another message type. In some cases, only part of a message type (for example, only the mandatory fields) is nested, but this “partial” nested message type is also considered to be nested. For example, SWIFT MT 195 could be used to request information about a SWIFT MT 100 (customer transfer). The SWIFT MT 100 (or at least its mandatory fields) is then nested in SWIFT MT 195.

nesting identifier. An identifier (a number from 2 to 255) that is used to access a nested message type.

network identifier. A single character that is placed before a message type to indicate which network is to be used to send the message; for example, **S** for SWIFT

network service access point (NSAP). The endpoint of a network connection used by the SWIFT transport layer.

NOPROMPT mode. One of two ways to display a message panel. NOPROMPT mode is only intended for experienced SWIFT Link users who are familiar with the structure of SWIFT messages. With NOPROMPT mode, only the SWIFT header, trailer, and pre-filled fields and their tags are displayed. Contrast with *PROMPT mode*.

NSAP. Network service access point.

nucleus server. A MERVA component that processes a service request as selected by the request queue handler. The service a nucleus server provides and the way it provides it is defined in the nucleus server table (DSLNSVT).

O

object. In MQSeries, objects define the properties of queue managers, queues, process definitions, and namelists.

occurrence. See *repeatable sequence*.

option. One or more characters added to a SWIFT field number to distinguish among different layouts for and meanings of the same field. For example, SWIFT field 60 can have an option F to identify a first opening balance, or M for an intermediate opening balance.

origin identifier (origin ID). A 34-byte field of the MERVA user file record. It indicates, in a MERVA and SWIFT Link installation that is shared by several banks, to which of these banks the user belongs. This lets the user work for that bank only.

OSN. Output sequence number.

OSN acknowledgment. A collective term for the various kinds of acknowledgments sent to the SWIFT network.

output message. A message that has been received from the SWIFT network. An output message has an output header.

P

P1. In MERVA Link, a peer-to-peer protocol used by cooperating message transfer processes (MTPs).

P2. In MERVA Link, a peer-to-peer protocol used by cooperating application support processes (ASPs).

P3. In MERVA Link, a peer-to-peer protocol used by cooperating command transfer processors (CTPs).

packet switched public data network (PSPDN). A public data network established and operated by network common carriers or telecommunication administrations for providing packet-switched data transmission.

panel. A formatted display on a display terminal. Each page of a message is displayed on a separate panel.

parallel processing. The simultaneous processing of units of work by several servers. The units of work can be either transactions or subdivisions of larger units of work.

parallel sysplex. A sysplex that uses one or more coupling facilities.

partner table (PT). In MERVA Link, the table that defines how messages are processed. It consists of a header and different entries, such as entries to specify the message-processing parameters of an ASP or MTP.

PCT. Program Control Table (of CICS).

PDE. Possible duplicate emission.

PDU. Protocol data unit.

PF key. Program-function key.

positional parameter. A parameter that must appear in a specified location relative to other parameters.

PREMIUM. The MERVA component used for SWIFT PREMIUM support.

process definition object. An MQSeries object that contains the definition of an MQSeries application. A queue manager uses the definitions contained in a process definition object when it works with trigger messages.

program-function key. A key on a display terminal keyboard to which a function (for example, a command) can be assigned. This lets you execute the function (enter the command) with a single keystroke.

PROMPT mode. One of two ways to display a message panel. PROMPT mode is intended for SWIFT Link users who are unfamiliar with the structure of SWIFT messages. With PROMPT mode, all the fields and tags are displayed for the SWIFT message. Contrast with *NOPROMPT mode*.

protocol data unit (PDU). In MERVA Link a PDU consists of a structured sequence of implicit and explicit data elements:

- Implicit data elements contain other data elements.
- Explicit data elements cannot contain any other data elements.

PSN. Public switched network.

PSPDN. Packet switched public data network.

PSTN. Public switched telephone network.

PT. Partner table.

PTT. A national post and telecommunication authority (post, telegraph, telephone).

Q

QDS. Queue data set.

QSN. Queue sequence number.

queue. (1) In MERVA, a logical subdivision of the MERVA queue data set used to store the messages associated with a MERVA message-processing function. A queue has the same name as the message-processing function with which it is associated. (2) In MQSeries, an object onto which message queuing applications can put messages, and from which they can get messages. A queue is owned and maintained by a queue manager. See also *request queue*.

queue element. A message and its related control information stored in a data record in the MERVA ESA Queue Data Set.

queue management. A MERVA service function that handles the storing of messages in, and the retrieval of messages from, the queues of message-processing functions.

queue manager. (1) An MQSeries system program that provides queueing services to applications. It provides an application programming interface so that programs can access messages on the queues that the queue manager owns. See also *local queue manager* and *remote queue manager*. (2) The MQSeries object that defines the attributes of a particular queue manager.

queue sequence number (QSN). A sequence number that is assigned to the messages stored in a logical queue by MERVA ESA queue management in ascending order. The QSN is always unique in a queue. It is reset to zero when the queue data set is formatted, or when a queue management restart is carried out and the queue is empty.

R

RACF. Resource Access Control Facility.

RBA. Relative byte address.

RC message. Recovered message; that is, an IP message that was copied from the control queue of an inoperable or closed ASP via the **recover** command.

ready queue. A MERVA queue used by SWIFT Link to collect SWIFT messages that are ready for sending to the SWIFT network.

remote queue. In MQSeries, a queue that belongs to a remote queue manager. Programs can put messages on remote queues, but they cannot get messages from remote queues. Contrast with *local queue*.

remote queue manager. In MQSeries, a queue manager is remote to a program if it is not the queue manager to which the program is connected.

repeatable sequence. A field or a group of fields that is contained more than once in a message. For example, if the SWIFT fields 20, 32, and 72 form a sequence, and if this sequence can be repeated up to 10 times in a message, each sequence of the fields 20, 32, and 72 would be an occurrence of the repeatable sequence.

In the TOF, the occurrences of a repeatable sequence are numbered in ascending order from 1 to 32767 and can be referred to using the occurrence number.

A repeatable sequence in a message may itself contain another repeatable sequence. To identify an occurrence within such a nested repeatable sequence, more than one occurrence number is necessary.

reply message. In MQSeries, a type of message used for replies to request messages.

reply-to queue. In MQSeries, the name of a queue to which the program that issued an MQPUT call wants a reply message or report message sent.

report message. In MQSeries, a type of message that gives information about another message. A report message usually indicates that the original message cannot be processed for some reason.

request message. In MQSeries, a type of message used for requesting a reply from another program.

request queue. The queue in which a service request is stored. It resides in main storage and consists of a set of request queue elements that are chained in different queues:

- Requests waiting to be processed
- Requests currently being processed
- Requests for which processing has finished

request queue handler (RQH). A MERVA ESA component that handles the queueing and scheduling of service requests. It controls the request processing of a nucleus server according to rules defined in the finite state machine.

Resource Access Control Facility (RACF). An IBM licensed program that provides for access control by identifying and verifying users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

retype verification. See *verification*.

routing. In MERVA, the passing of messages from one stage in a predefined processing path to the next stage.

RP. Regional processor.

RQH. Request queue handler.

RRDS. Relative record data set.

S

SAF. System Authorization Facility.

SCS. SNA character string

SCP. System control process.

SDI. Sequential data set input. A batch utility used to import messages from a sequential data set or a tape into MERVA ESA queues.

SDO. Sequential data set output. A batch utility used to export messages from a MERVA ESA queue to a sequential data set or a tape.

SDY. Sequential data set system printer. A batch utility used to print messages from a MERVA ESA queue.

service request. A type of request that is created and passed to the request queue handler whenever a nucleus server requires a service that is not currently available.

sequence number. A number assigned to each message exchanged between two nodes. The number is increased by one for each successive message. It starts from zero each time a new session is established.

sign off. To end a session with MERVA.

sign on. To start a session with MERVA.

single-system sysplex. A sysplex in which only one MVS system can be initialized as part of the sysplex. In a single-system sysplex, XCF provides XCF services on the system, but does not provide signalling services between MVS systems. A single-system sysplex requires an XCF couple data set. See also *multisystem sysplex*.

small queue element. A queue element that is smaller than the smaller of:

- The limiting value specified during the customization of MERVA
- 32KB

SMP/E. System Modification Program Extended.

SN. Session number.

SNA. Systems network architecture.

SNA character string. In SNA, a character string composed of EBCDIC controls, optionally mixed with user data, that is carried within a request or response unit.

SPA. Scratch pad area.

SQL. Structured Query Language.

SR-ASPDU. The status report application support PDU, which is used by MERVA Link for acknowledgment messages.

SSN. Select sequence number.

subfield. A subdivision of a field with a specific meaning. For example, the SWIFT field 32 has the subfields date, currency code, and amount. A field can have several subfield layouts depending on the way the field is used in a particular message.

SVC. (1) Switched Virtual Circuit. (2) Supervisor call instruction.

S.W.I.F.T. (1) Society for Worldwide Interbank Financial Telecommunication s.c. (2) The network provided and managed by the Society for Worldwide Interbank Financial Telecommunication s.c.

SWIFT address. Synonym for *bank identifier code*.

SWIFT Correspondents File. The file containing the bank identifier code (BIC), together with the name, postal address, and zip code of each financial institution in the BIC Directory.

SWIFT financial message. A message in one of the SWIFT categories 1 to 9 that you can send or receive via the SWIFT network. See *SWIFT input message* and *SWIFT output message*.

SWIFT header. The leading part of a message that contains the sender and receiver of the message, the message priority, and the type of message.

SWIFT input message. A SWIFT message with an input header to be sent to the SWIFT network.

SWIFT link. The MERVA ESA component used to link to the SWIFT network.

SWIFT network. Refers to the SWIFT network of the Society for Worldwide Interbank Financial Telecommunication (S.W.I.F.T.).

SWIFT output message. A SWIFT message with an output header coming from the SWIFT network.

SWIFT system message. A SWIFT general purpose application (GPA) message or a financial application (FIN) message in SWIFT category 0.

switched virtual circuit (SVC). An X.25 circuit that is dynamically established when needed. It is the X.25 equivalent of a switched line.

sysplex. One or more MVS systems that communicate and cooperate via special multisystem hardware components and software services.

System Authorization Facility (SAF). An MVS or VSE facility through which MERVA ESA communicates with an external security manager such as RACF (for MVS) or the basic security manager (for VSE).

System Control Process (SCP). A MERVA Link component that handles the transfer of MERVA ESA commands to a partner MERVA ESA system, and the receipt of the command response. It is associated with a system control process entry in the partner table.

System Modification Program Extended (SMP/E). A licensed program used to install software and software changes on MVS systems.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operating sequences for transmitting information units through, and for controlling the configuration and operation of, networks.

T

tag. A field identifier.

TCP/IP. Transmission Control Protocol/Internet Protocol.

Telex Correspondents File. A file that stores data about correspondents. When the user enters the corresponding nickname in a Telex message, the corresponding information in this file is automatically retrieved and entered into the Telex header area.

telex header area. The first part of the telex message. It contains control information for the telex network.

telex interface program (TXIP). A program that runs on a Telex front-end computer and provides a communication facility to connect MERVA ESA with the Telex network.

Telex Link. The MERVA ESA component used to link to the public telex network via a Telex substation.

Telex substation. A unit comprised of the following:

- Telex Interface Program
- A Telex front-end computer
- A Telex box

Terminal User Control Block (TUCB). A control block containing terminal-specific and user-specific information used for processing messages for display devices such as screen and printers.

test key. A key added to a telex message to ensure message integrity and authorized delivery. The test key is an integer value of up to 16 digits, calculated manually or by a test-key processing program using the significant information in the message, such as amounts, currency codes, and the message date.

test-key processing program. A program that automatically calculates and verifies a test key. The Telex Link supports panels for input of test-key-related data and an interface for a test-key processing program.

TFD. Terminal feature definitions table.

TID. Terminal identification. The first 9 characters of a bank identifier code (BIC).

TOF. Originally the abbreviation of *tokenized form*, the TOF is a storage area where messages are stored so that their fields can be accessed directly by their field names and other index information.

TP. Transaction program.

transaction. A specific set of input data that triggers the running of a specific process or job; for example, a message destined for an application program.

transaction code. In IMS and CICS, an alphanumeric code that calls an IMS message processing program or a CICS transaction. Transaction codes have 4 characters in CICS and up to 8 characters in IMS.

Transmission Control Protocol/Internet Protocol (TCP/IP). A set of communication protocols that support peer-to-peer connectivity functions for both local and wide area networks.

transmission queue. In MQSeries, a local queue on which prepared messages destined for a remote queue manager are temporarily stored.

trigger event. In MQSeries, an event (such as a message arriving on a queue) that causes a queue manager to create a trigger message on an initiation queue.

trigger message. In MQSeries, a message that contains information about the program that a trigger monitor is to start.

trigger monitor. In MQSeries, a continuously-running application that serves one or more initiation queues. When a trigger message arrives on an initiation queue, the trigger monitor retrieves the message. It uses the information in the trigger message to start a process that serves the queue on which a trigger event occurred.

triggering. In MQSeries, a facility that allows a queue manager to start an application automatically when predetermined conditions are satisfied.

TUCB. Terminal User Control Block.

TXIP. Telex interface program.

U

UMR. Unique message reference.

unique message reference (UMR). An optional feature of MERVA ESA that provides each message with a unique identifier the first time it is placed in a queue. It is composed of a MERVA ESA installation name, a sequence number, and a date and time stamp.

UNIT. A group of related literals or fields of an MCB definition, or both, enclosed by a DSLLUNIT and DSLLUEND macroinstruction.

UNIX System Services (USS). A component of OS/390, formerly called OpenEdition (OE), that creates a UNIX environment that conforms to the XPG4 UNIX 1995 specifications, and provides two open systems interfaces on the OS/390 operating system:

- An application program interface (API)
- An interactive shell interface

UN/EDIFACT. United Nations Standard for Electronic Data Interchange for Administration, Commerce and Transport.

USE. S.W.I.F.T. User Security Enhancements.

user file. A file containing information about all MERVA ESA users; for example, which functions each user is allowed to access. The user file is encrypted and can only be accessed by authorized persons.

user identification and verification. The acts of identifying and verifying a RACF-defined user to the system during logon or batch job processing. RACF identifies the user by the user ID and verifies the user by the password or operator identification card supplied during logon processing or the password supplied on a batch JOB statement.

USS. UNIX System Services.

V

verification. Checking to ensure that the contents of a message are correct. Two kinds of verification are:

- Visual verification: you read the message and confirm that you have done so
- Retype verification: you reenter the data to be verified

Virtual LU. An LU defined in MERVA Extended Connectivity for communication between MERVA and MERVA Extended Connectivity.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed and variable-length records on direct access devices. The records in a VSAM data set or file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry sequence), or by relative-record number.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

VSAM. Virtual Storage Access Method.

VTAM. Virtual Telecommunications Access Method (IBM licensed program).

W

Windows NT service. A type of Windows NT application that can run in the background of the Windows NT operating system even when no user is logged on. Typically, such a service has no user interaction and writes its output messages to the Windows NT event log.

X

X.25. An ISO standard for interface to packet switched communications services.

XCF. Abbreviation for *cross-system coupling facility*, which is a special logical partition that provides high-speed caching, list processing, and locking functions in a sysplex. XCF provides the MVS coupling services that allow authorized programs on MVS systems in a multisystem environment to communicate with (send data to and receive data from) authorized programs on other MVS systems.

XCF couple data sets. A data set that is created through the XCF couple data set format utility and, depending on its designated type, is shared by some or all of the MVS systems in a sysplex. It is accessed only by XCF and contains XCF-related data about the sysplex, systems, applications, groups, and members.

XCF group. The set of related members defined to SCF by a multisystem application in which members of the group can communicate with (send data to and receive data from) other members of the same group. All MERVA systems working together in a sysplex must pertain to the same XCF group.

XCF member. A specific function of a multisystem application that is defined to XCF and assigned to a group by the multisystem application. A member resides on one system in a sysplex and can use XCF services to communicate with other members of the same group.

Bibliography

MERVA ESA Publications

- *MERVA for ESA Version 4: Application Programming Interface Guide*, SH12-6374
- *MERVA for ESA Version 4: Advanced MERVA Link*, SH12-6390
- *MERVA for ESA Version 4: Concepts and Components*, SH12-6381
- *MERVA for ESA Version 4: Customization Guide*, SH12-6380
- *MERVA for ESA Version 4: Diagnosis Guide*, SH12-6382
- *MERVA for ESA Version 4: Installation Guide*, SH12-6378
- *MERVA for ESA Version 4: Licensed Program Specifications*, GH12-6373
- *MERVA for ESA Version 4: Macro Reference*, SH12-6377
- *MERVA for ESA Version 4: Messages and Codes*, SH12-6379
- *MERVA for ESA Version 4: Operations Guide*, SH12-6375
- *MERVA for ESA Version 4: System Programming Guide*, SH12-6366
- *MERVA for ESA Version 4: User's Guide*, SH12-6376

MERVA ESA Components Publications

- *MERVA Automatic Message Import/Export Facility: User's Guide*, SH12-6389
- *MERVA Connection/NT*, SH12-6339
- *MERVA Connection/400*, SH12-6340
- *MERVA Directory Services*, SH12-6367
- *MERVA Extended Connectivity: Installation and User's Guide*, SH12-6157
- *MERVA Message Processing Client for Windows NT: User's Guide*, SH12-6341
- *MERVA-MQI Attachment User's Guide*, SH12-6714
- *MERVA Traffic Reconciliation*, SH12-6392
- *MERVA USE: Administration Guide*, SH12-6338
- *MERVA USE & Branch for Windows NT: User's Guide*, SH12-6334
- *MERVA USE & Branch for Windows NT: Installation and Customization Guide*, SH12-6335
- *MERVA USE & Branch for Windows NT: Application Programming Guide*, SH12-6336
- *MERVA USE & Branch for Windows NT: Diagnosis Guide*, SH12-6337
- *MERVA USE & Branch for Windows NT: Migration Guide*, SH12-6393
- *MERVA USE & Branch for Windows NT: Installation and Customization Guide*, SH12-6335
- *MERVA Workstation Based Functions*, SH12-6383

Other IBM Publications

- *CICS/ESA Version 4.1 CICS-RACF Security Guide*, SC33-1185
- *CICS/ESA Version 4.1 Migration Guide*, GC33-1162
- *CICS/ESA Version 4.1 Release Guide*, GC33-1161
- *CICS/ESA Version 4.1 Resource Definition Guide*, SC33-1166
- *CICS/ESA Version 4.1 System Definition Guide*, SC33-1164
- *CICS TS Version 4.1 CICS-RACF Security Guide*, SC33-1701
- *CICS TS Version 4.1 DB2 Guide*, SC33-1939
- *CICS TS Version 4.1 Migration Guide*, GC33-1571
- *CICS TS Version 4.1 Release Guide*, GC33-1570
- *CICS TS Version 4.1 Resource Definition*, SC33-1684
- *CICS TS Version 4.1 System Definition Guide*, SC33-1682
- *DB2 for OS/390 Version 5.1 Command Reference*, SC26-8960
- *DB2 for OS/390 Version 5.1 SQL Reference*, SC26-8966
- *DB2 Server for VSE Version 5.1 DBS Utility*, SC09-2395
- *DB2 Server for VSE & VM Version 5.1 SQL Reference*, SC09-2404
- *DFSMS/MVS V1.4 Macro Instructions for Data Sets*, SC26-4913
- *CICS/VSE 2.3 Resource Definition (Macro)*, SC33-0709
- *CICS/VSE 2.3 Resource Definition (Online)*, SC33-0708

- *CICS/VSE 2.3 System Definition and Operations Guide*, SC33-0706
- *High Level Assembler Language Reference*, SC26-4940
- *IMS/ESA Version 5 Installation Vol. 1*, SC26-8023
- *IMS/ESA Version 5 Installation Vol. 2*, SC26-8024
- *IMS/ESA Version 5 Administration Guide: Database Manager*, SC26-8012
- *IMS/ESA Version 5 Administration Guide: System*, SC26-8013
- *IMS/ESA Version 5 Utilities Reference: Database Manager*, SC26-8034
- *IMS/ESA Version 5 Utilities Reference: System*, SC26-8035
- *IMS/ESA Version 5 Utilities Reference: Transaction Manager*, SC26-8022
- *MQSeries Application Programming Reference*, SC33-1673
- *MQSeries for VSE/ESA System Management Guide*, GC34-5364
- *MQSeries Command Reference*, SC33-1369
- *MVS/DFP 3.3: Macro Instructions for Data Sets*, SC26-4747
- *System Modification Program Extended Reference*, SC28-1107
- *Telex Interface Program: Program Description and Operations Manual*, SB11-8187
- *VSE/ESA Messages and Codes Vol. 1*, SC33-6698
- *VSE/ESA Messages and Codes Vol. 2*, SC33-6699
- *VSE/ESA System Control Statements*, SC33-6613

S.W.I.F.T. Publications

The following are published by the Society for Worldwide Interbank Financial Telecommunication, s.c., in La Hulpe, Belgium:

- *S.W.I.F.T. User Handbook*
- *S.W.I.F.T. Dictionary*
- *S.W.I.F.T. FIN Security Guide*
- *S.W.I.F.T. Card Readers User Guide*

Index

MERVA Requirement Request

Use the form overleaf to send us requirement requests for the MERVA product. Fill in the blank lines with the information that we need to evaluate and implement your request. Provide also information about your hardware and software environments and about the MERVA release levels used in your environment.

Provide a detailed description of your requirement. If you are requesting a new function, describe in full what you want that function to do. If you are requesting that a function be changed, briefly describe how the function works currently, followed by how you are requesting that it should work.

If you are a customer, provide us with the appropriate contacts in your organization to discuss the proposal and possible implementation alternatives.

If you are an IBM employee, include at least the name of one customer who has this requirement. Add the name and telephone number of the appropriate contacts in the customer's organization to discuss the proposal and possible implementation alternatives. If possible, send this requirement online to MERVAREQ at SDFVM1.

For comments on this book, use the form provided at the back of this publication.

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.

Send the fax to:

```
To: MERVA Development, Dept. 5640              Fax Number: +49-7031-16-4881
    Attention: Gerhard Stubbe                  Internet address:
                                               mervareq@de.ibm.com
    IBM Deutschland Entwicklung GmbH
    Schoenaicher Str. 220
    D-71032 Boeblingen
    Germany
```

MERVA Requirement Request

To: MERVA Development, Dept. 5640
Attention: Gerhard Strubbe

IBM Deutschland Entwicklung GmbH
Schoenaicher Str. 220
D-71032 Boeblingen Germany

Fax Number: +49-7031-16-4881
Internet address:
mervareq@de.ibm.com

Page 1 of ______

Customer's Name ____

Customer's Address ____

____

____

Customer's
Telephone/Fax ____

Contact Person at ____
Customer's Location
Telephone/Fax ____

MERVA
Version/Release ____

Operating System ____
Sub-System
Version/Release ____

Hardware ____

Requirement
Description ____

____

____

____

____

____

____

Expected Benefits ____

____

____

Readers' Comments — We'd Like to Hear from You

MERVA for ESA
Installation Guide
Version 4 Release 1

Publication No. SH12-6378-01

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

SH12-6378-01

Cut or Fold
Along Line

Fold and Tape

Please do not staple

Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland Entwicklung GmbH
Information Development, Dept. 0446
Schoenaicher Strasse 220
71032 Boeblingen
Germany

Fold and Tape

Please do not staple

Fold and Tape

Cut or Fold
Along Line

Program Number: 5648-B29

SH12-6378-01

Spine information:

IBM

MERVA for ESA

Installation Guide

Version 4
Release 1